

파이썬 프로그래밍

11. tkinter

❖ 수업 목표

- tkinter import를 할 수 있다.
- 윈도우 위젯 및 추가 위젯을 생성할 수 있다.
- Geometry manager(위젯 배치 관리자)를 사용할 수 있다.
- 이벤트 및 바인딩을 통해 위젯을 동작할 수 있다.

❖ 세부 목표

- 10.1 tkinter
- 10.2 위젯 생성(1)
- 10.3 Geometry manager
- 10.4 이벤트
- 10.5 바인딩
- 10.6 위젯 생성(2)

❖ tkinter 모듈

■ 파이썬 표준 라이브러리

```
import tkinter
```

■ 윈도우 위젯(widget, 작은 프로그램)(객체) 생성

- tkinter 라이브러리 클래스 중 윈도우 객체를 생성하는 Tk 클래스 사용
- Tk 클래스가 가장 상위층(Toplevel) 위젯을 생성
 - 이 위에 다른 위젯을 추가 가능

```
class tkinter.Tk(screenName=None, baseName=None, className='Tk', useTk=1)
```

The Tk class is instantiated without arguments. This creates a toplevel widget of Tk which usually is the main window of an application. Each instance has its own associated Tcl interpreter.

❖ 위젯 클래스 객체

- 최상위 객체인 윈도우 객체 위에 새 위젯 객체를 배치
- 윈도우 객체(부모 인스턴스)와 위젯 클래스 객체 연결 방법

위젯 클래스 객체=위젯(부모가 되는 인스턴스, option1=xxxx, option2=xxxx,)

❖ 위젯 클래스 배치

- geometry manager 인터페이스 메소드 중 하나인 pack()을 사용하여 윈도우 객체에 위젯 클래스 객체를 배치

```
위젯클래스객체.pack(option)  
위젯클래스객체.grid(option)  
위젯클래스객체.place(option)
```

❖ 이벤트와 바인딩

- 이벤트(Event, 사건)를 발생시키고 이벤트 내용은 함수 정의 및 호출

```
tkinter.Button(command=함수명)
```

2. 위젯 생성

❖ 윈도우 위젯 생성

- tkinter 최상위 위젯(윈도우 위젯) 클래스인 Tk 클래스를 사용
- 윈도우 객체 생성 코드

```
import tkinter
otk=tkinter.Tk() # -----> ①
obtn=tkinter.Button(otk, text="click") # --> ②
obtn.pack() # -----> ④
otk.mainloop()
```

```
from tkinter import *
otk=Tk()
obtn=Button(otk, text="click")
obtn.pack()
```



2. 위젯 생성

❖ 윈도우 위젯 생성

■ 윈도우 객체 생성 코드 설명

1. `import tkinter` : tkinter 모듈을 import 합니다.
2. `otk=tkinter.Tk()` : `Tk()` 함수를 사용하여 윈도우 창 객체를 생성하고, 그 객체를 'otk' 변수에 저장합니다.
3. `obtn=tkinter.Button(otk, text="click")` : `Button()` 함수를 사용하여 'otk' 객체 내에 버튼 객체를 생성하고, 그 버튼 객체를 'obtn' 변수에 저장합니다. 'text' 매개변수를 사용하여 버튼의 텍스트를 "click me"로 지정합니다.
4. `obtn.pack()` : `pack()` 함수를 사용하여 'obtn' 버튼 객체를 'otk' 윈도우 창에 배치합니다.
5. `otk.mainloop()` : `mainloop()` 함수를 사용하여 'otk' 윈도우 창이 계속해서 실행합니다. 윈도우 창이 종료될 때까지 프로그램의 실행 흐름을 막습니다.

2. 위젯 생성

❖ 윈도우 위젯 생성

■ Tk 클래스

syntax : geometry 메소드

```
geometry("가로길이x세로길이+윈도우창왼쪽x축 좌표+윈도우창왼쪽y축좌표")
```

- **문법 추가 설명**

- x : 알파벳 소문자 x
- + : 연산자 +
- 숫자와 각 기호들 사이에 띄어쓰기 없이 표시

2. 위젯 생성

❖ 윈도우 위젯 생성

■ Tk 클래스

- geometry 메소드 예시 코드

syntax : geometry()

```
import tkinter
otk = tkinter.Tk()
otk.geometry("100x150")
obtn = tkinter.Button(otk, text="click")
obtn.pack()
otk.mainloop()
```

```
import tkinter
otk = tkinter.Tk()
otk.geometry("100x150+400+400")
obtn = tkinter.Button(otk, text="click")
obtn.pack()
otk.mainloop()
```

2. 위젯 생성

❖ 윈도우 위젯 생성

■ 함수 호출

syntax : 함수

command = 함수명

```
def hello():  
    print("hello there")  
from tkinter import *  
otk=Tk()  
obtn=Button(otk, text="click me", command=hello) # ---> ①  
obtn.pack()
```

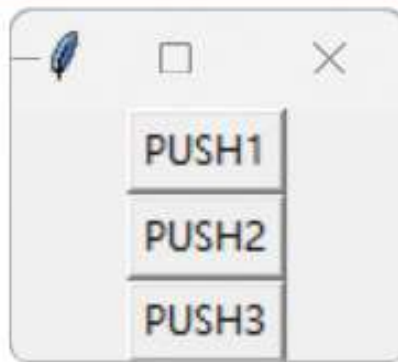
2. 위젯 생성

❖ 버튼 위젯 생성

■ 버튼 3개 생성

syntax : Button 클래스 & pack() 메소드

```
from tkinter import *  
oroot = Tk()  
obutton1 = Button(oroot, text="PUSH1")  
obutton2 = Button(oroot, text="PUSH2")  
obutton3 = Button(oroot, text="PUSH3")  
obutton1.pack()  
obutton2.pack()  
obutton3.pack()  
oroot.mainloop()
```



2. 위젯 생성

❖ 레이블 위젯 생성

- 레이블(Label) 클래스 : 텍스트를 화면에 출력하기 위해 사용

syntax : Label 클래스 & pack() 메소드

```
from tkinter import *  
oroot = Tk()  
olabel1 = Label(oroot, text="적", bg="red", width = 20)  
olabel2 = Label(oroot, text="녹", bg="green", width=20)  
olabel3 = Label(oroot, text="파", bg="blue", width=20)  
olabel1.pack()  
olabel2.pack()  
olabel3.pack()  
oroot.mainloop()
```



2. 위젯 생성

❖ 버튼 및 레이블 위젯 매개변수 정의

■ 필요시 검색 활용

1. text : 버튼에 표시될 텍스트를 지정합니다.
2. font : 버튼의 글꼴을 지정합니다.
3. fg(foreground) : 버튼의 텍스트 색상을 지정합니다.
4. bg(background) : 버튼의 배경 색상을 지정합니다.
5. width : 버튼의 폭을 지정합니다.
6. height : 버튼의 높이를 지정합니다.
7. command : 버튼을 클릭했을 때 실행할 함수를 지정합니다.
8. state : 버튼의 상태를 지정합니다. 'normal', 'active', 'disabled' 등이 있습니다.
9. padx : 버튼의 좌우 여백을 지정합니다.
10. pady : 버튼의 상하 여백을 지정합니다.
11. relief : 버튼의 외곽선 모양을 지정합니다. 'flat', 'raised', 'sunken', 'groove', 'ridge' 등이 있습니다.
12. image : 버튼에 표시될 이미지를 지정합니다.
13. compound : 버튼의 이미지와 텍스트를 어떻게 결합할지 지정합니다. 'none', 'top', 'bottom', 'left', 'right', 'center' 등이 있습니다.

이 외에도 다양한 옵션이 존재합니다. 이 중에서도 가장 자주 사용되는 옵션은 text, command, fg, bg 등입니다. 이러한 옵션을 조합하여 버튼 위젯을 원하는 모양으로 꾸밀 수 있습니다. 또한, command 옵션을 활용하여 버튼을 클릭했을 때 실행할 함수를 지정할 수 있습니다.

1. text : 라벨에 표시될 텍스트를 지정합니다.
2. font : 라벨의 글꼴을 지정합니다.
3. fg(foreground) : 라벨의 텍스트 색상을 지정합니다.
4. bg(background) : 라벨의 배경 색상을 지정합니다.
5. width : 라벨의 폭을 지정합니다.
6. height : 라벨의 높이를 지정합니다.
7. anchor : 라벨의 텍스트나 이미지를 어디에 맞출지 지정합니다. 'n', 's', 'e', 'w' 및 이들을 조합한 문자열로 지정할 수 있습니다.
8. padx : 라벨의 좌우 여백을 지정합니다.
9. pady : 라벨의 상하 여백을 지정합니다.
10. relief : 라벨의 외곽선 모양을 지정합니다. 'flat', 'raised', 'sunken', 'groove', 'ridge' 등이 있습니다.
11. image : 라벨에 표시될 이미지를 지정합니다.
12. compound : 라벨의 이미지와 텍스트를 어떻게 결합할지 지정합니다. 'none', 'top', 'bottom', 'left', 'right', 'center' 등이 있습니다.
13. justify : 라벨의 텍스트가 여러 줄일 경우 어떻게 정렬할지 지정합니다. 'left', 'right', 'center' 등이 있습니다.
14. wraplength : 라벨의 텍스트가 한 줄을 넘어갈 경우, 텍스트를 감싸는 너비를 지정합니다.

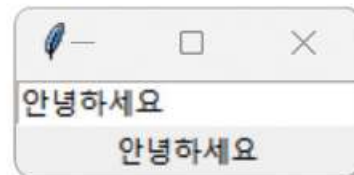
이 외에도 다양한 옵션이 존재합니다. 이 중에서도 가장 자주 사용되는 옵션은 text, fg, bg, font 등입니다. 이러한 옵션을 조합하여 라벨 위젯을 원하는 모양으로 꾸밀 수 있습니다.

2. 위젯 생성

❖ Entry 위젯 생성

■ 문자 동적 입력 위젯

```
import tkinter
oroot=tkinter.Tk()
ostring=tkinter.StringVar() # -----> ①
oentry=tkinter.Entry(oroot, textvariable=ostring) # ---> ②
oentry.pack()
olabel=tkinter.Label(oroot, textvariable=ostring)
olabel.pack()
oroot.mainloop()
```



- ① **StringVar()**는 실행 시 변수 타입이 자동 결정(동적 할당)
- ② 문자열 변수 선언 후, **textvariable** 옵션 값 **ostring** 할당
 - Entry 객체에 입력되는 데이터가 **ostring** 변수에 저장

3. Geometry manager

❖ Geometry manager : 배치 관리자

■ 윈도우 위젯에 하위 위젯을 위치시키는 메소드

종류	위치 옵션	위치 옵션 값	예시
pack()	side	"left", "right", "top", "bottom"	pack(side='left')
grid()	row, column	행렬 위치를 사용하면 된다. 실행하면 column=4라고 입력해도 상대적 위치인 2열에 위치한다.	grid(row=1, column=0)
place()	x, y	윈도우 창 내에서 절대 x, y 위치를 의미한다.	place(x=10, y=60)

3. Geometry manager

❖ 위젯 배치 메서드

- **grid() : 행열 기준 배치**

syntax : grid()

```
import tkinter
oroot = tkinter.Tk()
oroot.geometry("200x100")
obutton1 = tkinter.Button(oroot, text="PUSH1")
obutton2 = tkinter.Button(oroot, text="PUSH2")
obutton3 = tkinter.Button(oroot, text="PUSH3")
obutton1.grid(row=1, column=0)
obutton2.grid(row=1, column=1)
obutton3.grid(row=0, column=4)
oroot.mainloop()
```



3. Geometry manager

❖ 위젯 배치 메서드

- **pack()** : 좌/우/상/하 배치

syntax : pack()

```
import tkinter
oroot = tkinter.Tk()
oroot.geometry("200x100+600+300")
obutton1 = tkinter.Button(oroot, text="PUSH1")
obutton2 = tkinter.Button(oroot, text="PUSH2")
obutton3 = tkinter.Button(oroot, text="PUSH3")
obutton1.pack(side='left')
obutton2.pack(side='right')
obutton3.pack(side='top')
oroot.mainloop()
```



3. Geometry manager

❖ 위젯 배치 메서드

- **place()** : 좌표 기준 배치

syntax : place()

```
import tkinter
oroot = tkinter.Tk()
oroot.geometry("200x100")
obutton1 = tkinter.Button(oroot, text="PUSH1")
obutton2 = tkinter.Button(oroot, text="PUSH2")
obutton3 = tkinter.Button(oroot, text="PUSH3")
obutton1.place(x=10, y=60)
obutton2.place(x=140, y=60)
obutton3.place(x=80, y=10)
oroot.mainloop()
```



❖ 이벤트(event, 사건)

- 프로그램에 의해 감지되고 처리될 수 있는 동작이나 사건
- 버튼 클릭 시 이벤트가 발생하는 함수 정의
 - 주문 버튼 생성

```
import tkinter
def order():
    print("주문하세요")
root=tkinter.Tk()
btn=tkinter.Button(root,text="주문", command=order)
btn.pack()
root.mainloop()
```

주문하세요



❖ 바인딩(binding, 묶다)

- 이벤트 발생 시, 동작 사항을 연결하는 것
- 버튼 클릭 시, 옵션 설정을 통해 이벤트가 발생하는 함수 호출
 - 시작 버튼 생성

```
import tkinter
def msg():
    print("start tkinter")
root=tkinter.Tk()
mlabel=tkinter.Label(root, text="시작레이블")
mlabel.pack(side='top')

mbutton=tkinter.Button(root, text="시작버튼", command=msg)
mbutton.pack(side='bottom')
root.mainloop()
```



6. 위젯 생성(2)

❖ 라디오 버튼

- 선택 범주 중 하나만 선택 가능



6. 위젯 생성(2)

❖ 라디오 버튼

■ 라디오 버튼 코드

```
import tkinter
oroot=tkinter.Tk()
radio_value=tkinter.IntVar() # -----> ①
radio_value.set(1) # -----> ②
lunch={0: "A런치", 1:"B런치", 2:"C런치"}
orb1=tkinter.Radiobutton(oroot, text=lunch[0],variable=radio_value, value=0)
# ----> ③

orb1.pack()
orb2=tkinter.Radiobutton(oroot, text=lunch[1], variable=radio_value, value=1)
orb2.pack()
orb3=tkinter.Radiobutton(oroot, text=lunch[2], variable=radio_value, value=2)
orb3.pack()
def buy():
    value=radio_value.get() # -----> ④
    print(lunch[value])

obutton=tkinter.Button(oroot, text="주문", command=buy)
obutton.pack()
oroot.mainloop()
```

〈화면 출력〉

C런치

6. 위젯 생성(2)

❖ 라디오 버튼

- 라디오 버튼 코드 설명
- **Radiobutton(master, text, variable, value) 매개변수**

1. master : 라디오 버튼이 배치될 윈도우나 프레임을 지정합니다. 필수 인수입니다.
2. text : 라디오 버튼 옆에 표시될 레이블의 텍스트를 지정합니다. 기본값은 빈 문자열("")입니다.
3. variable : 라디오 버튼이 선택되었을 때 선택된 값을 저장하기 위한 변수를 지정합니다. 모든 라디오 버튼은 같은 변수를 사용해야 하며, 이 변수는 IntVar(), DoubleVar(), StringVar(), BooleanVar() 중 하나의 인스턴스여야 합니다. 필수 인수입니다.
4. value : 라디오 버튼이 선택되었을 때 variable 변수에 저장될 값입니다. 기본값은 0입니다.
5. command : 라디오 버튼이 선택되었을 때 실행될 함수를 지정합니다.

6. 위젯 생성(2)

❖ 라디오 버튼

■ 라디오 버튼 코드 설명

- Radiobutton(master, text, variable, value) 매개변수

1. master는 위의 코드 oroot에 해당하는 부분이다.
2. text는 라디오 버튼 오른쪽에 나타난 문자열을 표시한다.
3. variable 부분부터가 좀 더 설명이 필요한 부분이다. 즉, 예제에 3개의 라디오 버튼이 있는데 이 중 어느 버튼이 클릭되어 있는지 정보를 저장할 변수명을 지정해 주는 것이다. 예제에서는 radio_value라는 변수에 저장한다고 지정한다.
4. value는 variable 인수가 지정한 변수 radio_value에 저장될 데이터를 지정하는 인수이다. 즉, 세 번째 라디오 버튼을 클릭하면 value에 지정되어 있는 정수 2가 radio_value 변수에 저장된다.

6. 위젯 생성(2)

❖ 라디오 버튼

- 라디오 버튼 코드 설명
- IntVar() 클래스
 - 정수 값을 저장하는 변수 생성

tkinter.IntVar(), tkinter.StringVar(), 그리고 tkinter.BooleanVar()은 파이썬의 tkinter 모듈에서 제공하는 클래스로, GUI (그래픽 사용자 인터페이스) 어플리케이션에서 사용할 수 있는 변수를 나타냅니다.

- tkinter.IntVar()는 정수 값을 저장하는 변수를 생성합니다. 이 변수는 Checkbutton, Radiobutton, 그리고 Scale 등과 같은 위젯과 연결될 수 있습니다.
- tkinter.StringVar()는 문자열 값을 저장하는 변수를 생성합니다. 이 변수는 Entry, Label, 그리고 OptionMenu 등과 같은 위젯과 연결될 수 있습니다.
- tkinter.BooleanVar()는 불리언 값을 저장하는 변수(참(True)) 또는 거짓(False))을 생성합니다. 이 변수는 Checkbutton 등과 같은 위젯과 연결될 수 있습니다.

이러한 변수들은 GUI에서 사용자가 입력한 값을 저장하고, 변수 값이 변경될 때 동적으로 위젯에 표시되는 값을 업데이트할 수 있도록 합니다. 이러한 변수들을 사용하여 Python에서 상호작용하는 GUI 애플리케이션을 만들 수 있습니다.

6. 위젯 생성(2)

❖ 라디오 버튼

- 라디오 버튼 코드 설명
- `get()` 메서드
 - 동적 변수에 저장된 값을 가져옴
- `set()` 메서드
 - 동적 변수에 값 설정

`get()` 메서드와 `set()` 메서드는 `tkinter` 모듈의 동적 변수 (`IntVar()`, `StringVar()`, `BooleanVar()` 등)에서 사용할 수 있는 메서드입니다.

`get()` 메서드는 동적 변수에 저장된 값을 가져오는데 사용됩니다. 예를 들어, `my_var = tkinter.StringVar()` 와 같이 문자열 값을 저장하는 `StringVar()` 인스턴스를 생성한 후, `my_var.get()`을 호출하면 이 변수에 저장된 값이 반환됩니다. 이 메서드를 사용하여 사용자가 입력한 값을 가져올 수 있습니다.

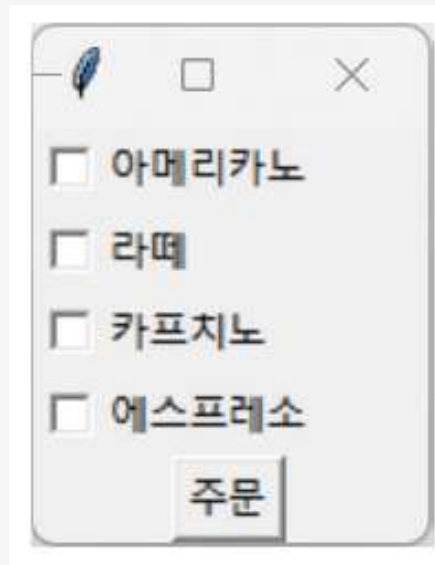
`set()` 메서드는 동적 변수에 값을 설정하는데 사용됩니다. 예를 들어, `my_var.set("Hello, world!")`와 같이 `set()` 메서드를 사용하여 `my_var`에 "Hello, world!"라는 문자열 값을 설정할 수 있습니다. 이 메서드를 사용하여 GUI 어플리케이션의 위젯에 표시되는 값을 업데이트할 수 있습니다.

따라서, `get()` 메서드와 `set()` 메서드는 동적 변수의 값을 가져오거나 설정하여 GUI 어플리케이션에서 사용자가 입력한 값을 저장하거나, 그리고 동적 변수의 값을 업데이트하여 어플리케이션의 상태를 변경하는 데 사용됩니다.

6. 위젯 생성(2)

❖ 체크 버튼(박스)

- 주어진 group(범주)중 여러 개 선택 가능



6. 위젯 생성(2)

❖ 체크 버튼(박스)

■ 체크 버튼 코드

```
import tkinter
oroot=tkinter.Tk()
coffee={0:"아메리카노", 1:"라떼", 2:"카프치노", 3:"에스프레소"}
check_value={}
for i in range(len(coffee)):
    check_value[i]=tkinter.BooleanVar()
    ocheckboxbutton=tkinter.Checkbutton(oroot,variable=check_value[i],
text=coffee[i])
    ocheckboxbutton.pack(anchor="w")

def buy():
    for i in check_value:
        if check_value[i].get()==True:
            print(coffee[i])

tkinter.Button(oroot, text="주문", command=buy).pack()
oroot.mainloop()
```

6. 위젯 생성(2)

❖ 체크 버튼(박스)

■ 체크 버튼 코드 설명

- **Checkbox(master, text, variable) 매개변수**

tkinter.Checkbutton() 함수는 체크박스를 생성하는 데 사용되는 Tkinter 위젯입니다. 체크박스는 사용자가 선택/해제할 수 있는 옵션을 제공하는 데 사용됩니다. tkinter.Checkbutton() 함수는 다음과 같은 인수를 받습니다:

- master : 이 체크박스 위젯의 부모 위젯입니다.
- text : 체크박스 옆에 표시되는 레이블 텍스트입니다.
- variable : 체크박스의 상태를 나타내는 Tkinter 변수입니다. 체크박스가 선택되면 이 변수의 값이 1로 설정되며, 해제되면 0으로 설정됩니다. 이 인수는 필수적입니다.
- onvalue : variable 변수가 체크되었을 때의 값입니다. 기본값은 1입니다.
- offvalue : variable 변수가 해제되었을 때의 값입니다. 기본값은 0입니다.

6. 위젯 생성(2)

❖ 체크 버튼(박스)

■ 체크 버튼 코드 설명

- `pack(anchor= "w")`
 - `anchor` : 위젯 내 표시 대상(문자나 이미지)의 위치를 지정하는 특성
 - » 매개변수 : “n” , “s” , “e” , “w” , “ne” , “nw” , “se” , “sw” , “center” 중 하나의 값
 - » “n” , “s” , “e” , “w” 는 각각 north(북), south(남), east(동), west(서)를 의미



6. 위젯 생성(2)

❖ 이미지 추가

■ 레이블에 이미지 추가

- ① 위젯 이름을 'img1' 으로 이미지 위젯 생성
 - » 사용할 이미지는 현재 프로그램 파일과 동일 폴더에 저장
- ② 레이블에 text 대신 image 추가
- ③ 저장한 'pizzasb2.png' 이미지는 화면에 표시
 - » 이미지의 중심 좌표가 place(x=-20, y=-20)

```
import tkinter
oroot=tkinter.Tk()
oroot.geometry("200x100")
img1=tkinter.PhotoImage(file='pizzasb2.png') # -> ①
img_label=tkinter.Label(oroot, image=img1) # ---> ②
img_label.place(x=-20, y=-20) # -----> ③
obutton1=tkinter.Button(oroot, text="PUSH1")
obutton2=tkinter.Button(oroot, text="PUSH2")
obutton3=tkinter.Button(oroot, text="PUSH3")
obutton1.place(x=10, y=60)
obutton2.place(x=140, y=60)
obutton3.place(x=80, y=10)
oroot.mainloop()
```



6. 위젯 생성(2)

❖ 옵션 메뉴

- 여러 항목들 중 하나를 선택할 수 있도록 하는 위젯
- 옵션 메뉴 선택 코드

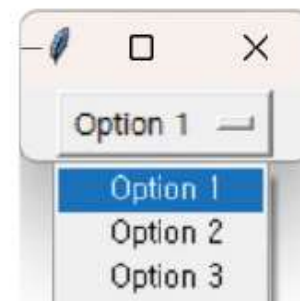
```
import tkinter as tk
root=tk.Tk()

# 옵션 리스트
options_list=['Option 1', 'Option 2', 'Option 3']

# 선택한 옵션을 저장할 변수
selected_option=tk.StringVar()

# 초기 선택 옵션
selected_option.set(options_list[0])

# OptionMenu 생성
option_menu=tk.OptionMenu(root, selected_option,
*options_list)
option_menu.pack()
root.mainloop()
```



6. 위젯 생성(2)

❖ 옵션 메뉴

■ 옵션 메뉴 선택 코드



OptionMenu 위젯에 사용할 옵션 리스트를 생성합니다.

```
options_list=['Option 1', 'Option 2 ', 'Option 3']
```

선택한 옵션을 저장할 변수를 생성합니다.

```
selected_option=tk.StringVar()
```

변수 'selected_option'의 초기 값을 옵션 리스트의 첫 번째 값으로 설정합니다.

```
selected_option.set(options_list[0])
```

'OptionMenu' 위젯을 생성합니다.

```
option_menu=tk.OptionMenu(root, selected_option, *options_list)
```

6. 위젯 생성(2)

❖ 옵션 메뉴

■ 옵션 메뉴 선택 코드

- `*option_list`



`*options_list`에 왜 `*`가 붙었는지 설명해줘.



`*` 기호는 함수 호출 시 리스트의 요소를 개별 인자로 전달하기 위한 파이썬의 언패킹(unpacking) 기능입니다.

위 예제에서 'OptionMenu' 생성자의 인자 중 `*options_list`는 'options_list' 리스트의 모든 요소를 개별 인자로 전달하기 위해 사용됩니다. 따라서 `*options_list`를 사용하면 'OptionMenu' 생성자는 다음과 같은 인자를 받게 됩니다.

```
option_menu=tk.OptionMenu(root, selected_option, 'Option 1', 'Option 2',
                           'Option 3')
```

만약 `*`를 사용하지 않고 'options_list' 리스트 자체를 전달하면 다음과 같은 결과가 발생합니다.

```
option_menu=tk.OptionMenu(root, selected_option, options_list)
```

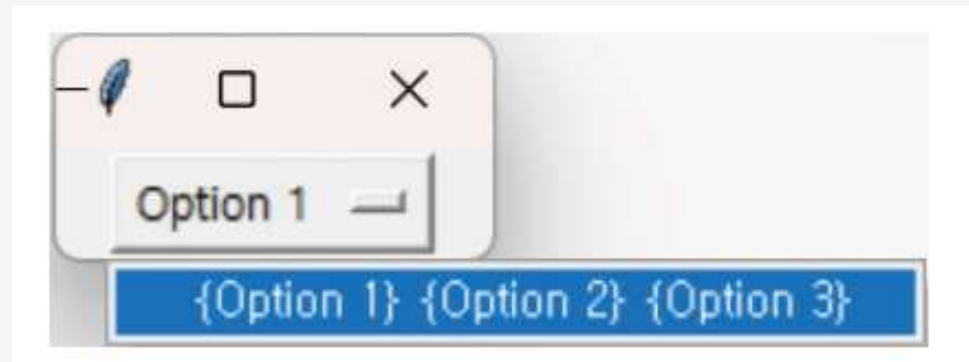
위 코드는 'OptionMenu' 생성자의 인자 중 세 번째 인자(`*options_list`)에 'options_list' 리스트 자체를 전달하므로, 'OptionMenu' 위젯의 옵션 리스트가 리스트 자체로 표시되어 제대로 작동하지 않습니다. 따라서 `*` 기호를 사용하여 리스트의 요소를 개별 인자로 전달해야 합니다.

6. 위젯 생성(2)

❖ 옵션 메뉴

■ 옵션 메뉴 선택 코드

- `option _menu = tk.OptionMenu(root, selected _option,options _list)`로 변경 실행 시, 다음 결과 실행



❖ 과제

- 1. tkinter 모듈의 구성을 import 하며 작성하기
- 2. 기본 위젯을 생성하고 객체를 생성하기
- 3. 위젯 객체 배치를 위한 3가지 방법 코드로 작성하기
- 4. 이벤트 발생을 통해 함수 동작 코드 작성하기
- 5. 위젯 객체 생성시 옵션 검색하며 작성하기

❖ 다음 수업 내용

- 파일 처리
 - 파일 입력 / 출력 / 읽기 / 쓰기, 파일 열기/닫기