으뜸 파이썬

9강 GUI와 실행파일

GUI 프로젝트

- Tkinter: built-in

```
import tkinter as tk


window = tk.Tk()
window.title("My First GUI")
label = tk.Label(window, text="Hello,
World!")
label.pack()
window.mainloop()
```

Tkinter 모듈

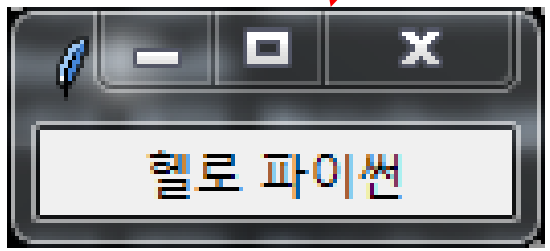
- 그래픽 사용자 인터페이스Graphical User Interface:GUI
  - 컴퓨터와 같은 전자기기의 디스플레이 장치에 시각적인 아이콘과 이미지로 정보를 표시하며, 마우스나 터치패드와 같은 기기를 이용하여 사용자가 조작할 수 있는 인터페이스를 말함
- Tkinter 모듈
  - 그래픽 기반의 사용자 인터페이스 프로그램을 개발하는데 사용되는 모듈
  - 티-케이-인터라고 읽는다. Tk라는 GUI 도구를 기반으로 작성되었다

Tkinter를 이용한 간단한 실습 코드

tkinter_hello.py

```
from tkinter import *  # tkinter 모듈내의 모든 클래스와 함수를 가져옴


window = Tk() # tkinter 모듈내의 Tk라는 윈도창 객체를 생성
label = Label(window, text = '헬로 파이썬') # 레이블 객체를 생성
label.pack()  # 컨테이너에 모듈을 위치시킴


window.mainloop()
```

레이블 객체를 놓을 수 있는 그릇 : 컨테이너라고 함



```
from tkinter import *
```

- tkinter라는 모듈내에 정의된 모든 클래스와 함수, 상수를 이 코드 안으로 가져오는 역할

```
window = Tk()
```

- Tk라는 윈도우 객체를 생성(다른 GUI 요소를 포함하는 컨테이너)

```
label = Label(window, text='헬로 파이썬')
```

- 위젯widget
  - 미리 만들어진 제어 가능한 요소로 레이블, 버튼, 체크박스 등
- 부모 컨테이너parent containter
  - 위젯 생성시의 첫 번째 인자
  - 이 레이블이 표시되는 윈도우나 캔버스가 됨

label.pack()

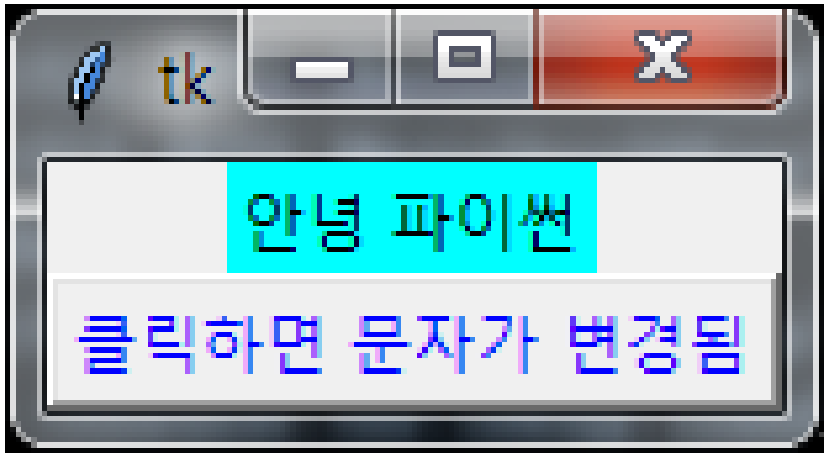
- pack()
  - 컨테이너에 레이블을 위치

window.mainloop()

- 이벤트 루프event loop를 생성
- 이벤트event란 사용자가 마우스를 이동해서 버튼이나 체크 상자를 클릭하거나 키보드에 입력을 하는 행위

버튼과 이벤트 처리

- 버튼과 레이블을 가지고 버튼을 클릭할 때마다 서로 다른 레이블을 표시하는 프로그램을 만들어 보기

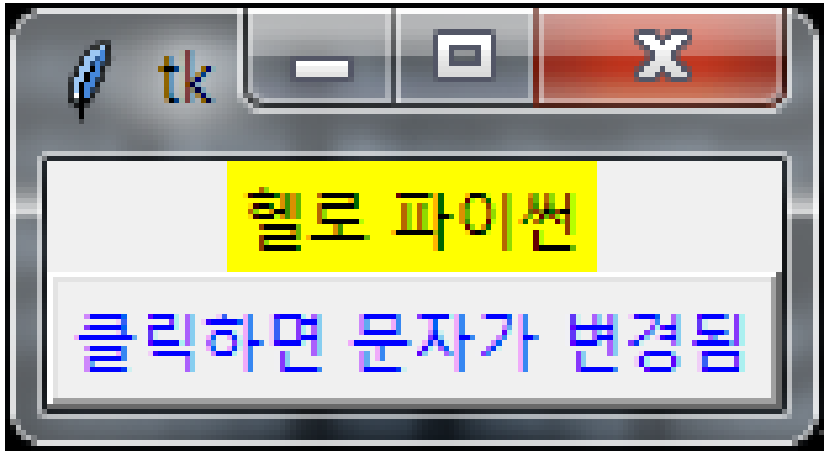


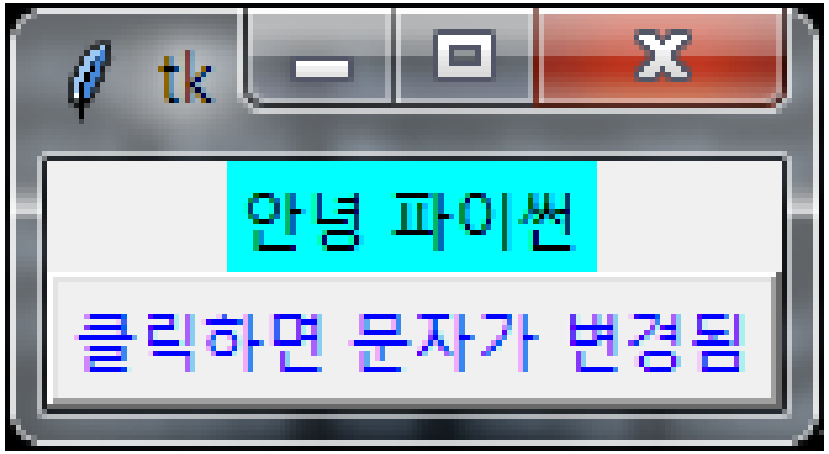
Tkinter를 이용한 간단한 실습 코드

tkinter_change_label.py

```
from tkinter import *
def change_label():
    if label.cget("text") == '헬로 파이썬':
        label.config(text = '안녕 파이썬')
        label.config(bg = 'cyan')
    else:
        label.config(text = '헬로 파이썬')
        label.config(bg = 'yellow')


window = Tk()
label = Label(window, text = '헬로 파이썬', bg = 'yellow')
label.pack()
btn = Button(window, text = '클릭하면 문자가 변경됨', fg = 'blue', command = change_label)
btn.pack()
window.mainloop()
```

- ‘클릭하면 문자가 변경됨’ 버튼 클릭시 화면상의 레이블 텍스트와 배경 색상이 변경됨





엔트리 객체를 이용한 사용자 입력 받기

- 엔트리 객체는 사용자의 키보드 입력을 문자열로 받을 때에 유용하다.
- 엔트리 객체를 생성하는 것은 레이블이나 버튼을 생성하는 것과 유사하다
- 아래와 같은 방식으로 엔트리 객체를 생성하면 윈도의 하위 노드로 엔트리가 생성되며, 폭은 50이 된다

```
input_entry = Entry(window, width=50)
```

체질량지수를 계산하기 위해 필요한 인터페이스 요소를 배치하는 코드

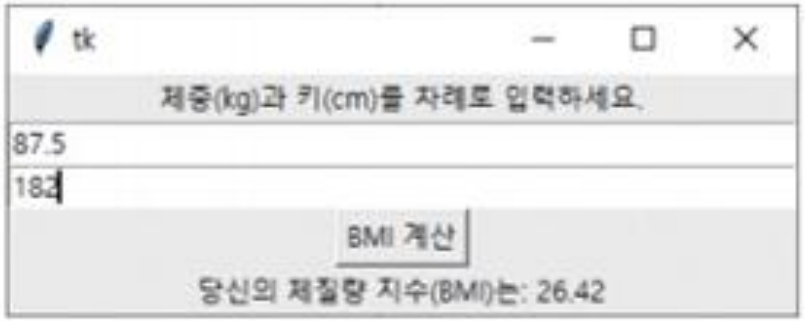
bmi_input_interface.py

```
from tkinter import *


window = Tk()
label = Label(window, text='체중(kg)과 키(cm)를 차례로 입력하세요.')
label.pack()
weight = Entry(window, width = 50)
weight.pack()
height = Entry(window, width = 50)
height.pack()
button = Button(window, text = 'BMI 계산')
button.pack()
result = Label(window, text='당신의 체질량 지수(BMI)는: ')
result.pack()


window.mainloop()
```

체질량지수를 계산기

bmi_calculator.py

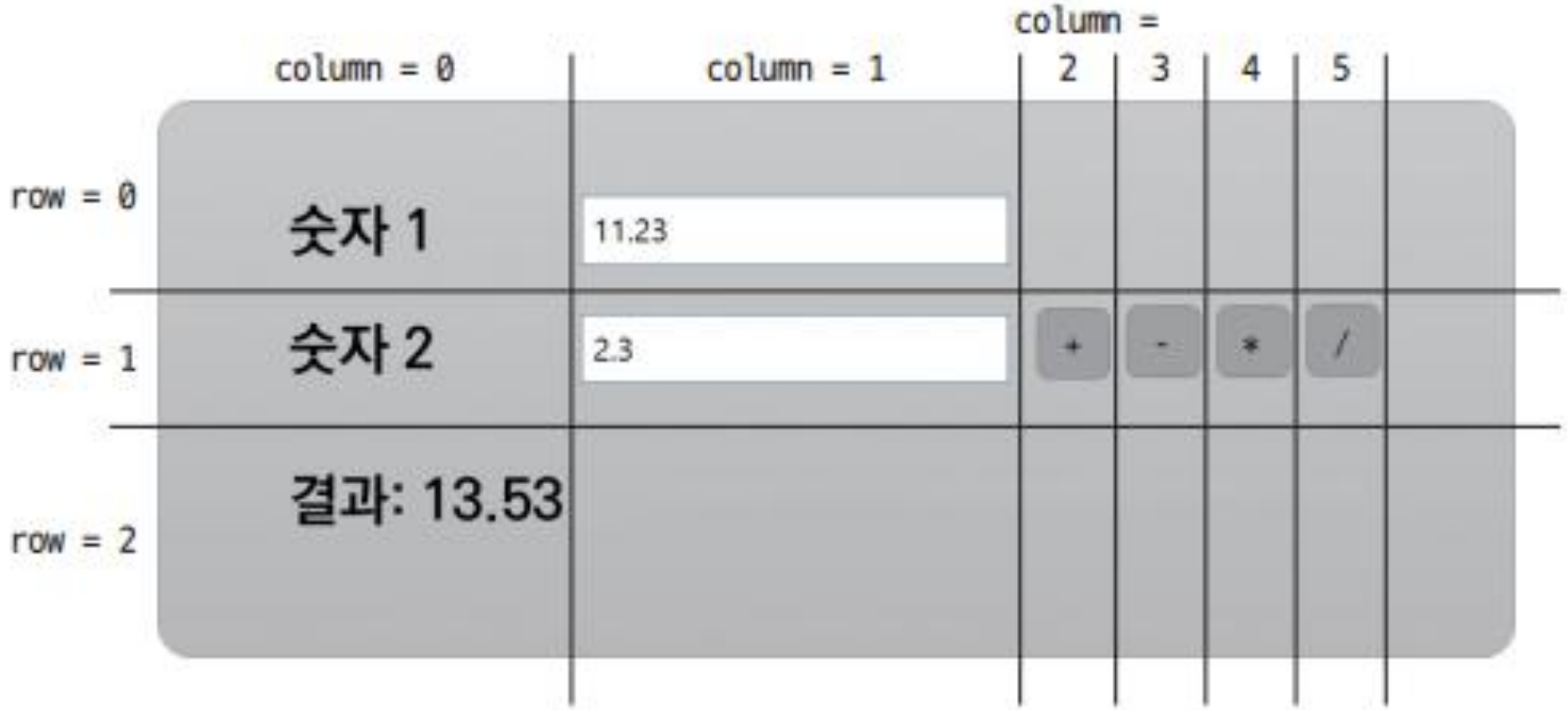
```
from tkinter import *
def bmi_compute():                    # 체질량 계산 함수
    w = float(weight.get())           # 입력된 체중 값을 실수로 변환
    h = float(height.get())/100.0     # cm로 입력된 키를 m 단위 환산
    bmi = w/(h*h)                     # 체질량 값을 계산함
    answer = '당신의 체질량 지수(BMI)는: {:4.2f}'.format(bmi)
    result.config(text=answer)        # 체질량 값을 result 레이블에 적용


window = Tk()
label = Label(window, text='체중(kg)과 키(cm)를 차례로 입력하세요.')
label.pack()
weight = Entry(window, width = 50)
weight.pack()
height = Entry(window, width = 50)
height.pack()
button = Button(window, text = 'BMI 계산', command=bmi_compute)
button.pack()
result = Label(window, text='당신의 체질량 지수(BMI)는: ')
result.pack()
window.mainloop()
```



계산기 만들기

- 이 계산기는 두수를 입력하는 엔트리 박스와 레이블, 그리고 덧셈, 뺄셈, 곱셈, 나눗셈 연산을 처리하는 4개의 버튼 그리고 결과를 출력하는 부분으로 구성할 예정이다

- 인터페이스 배치를 위해 각 인터페이스 요소를 격자로 구분하여 배열한 다면 다음과 같이 표현할 수 있다. 이때 + 버튼의 위치를 예를 들어보면 row = 1, column = 2 라고 표현할 수 있다.



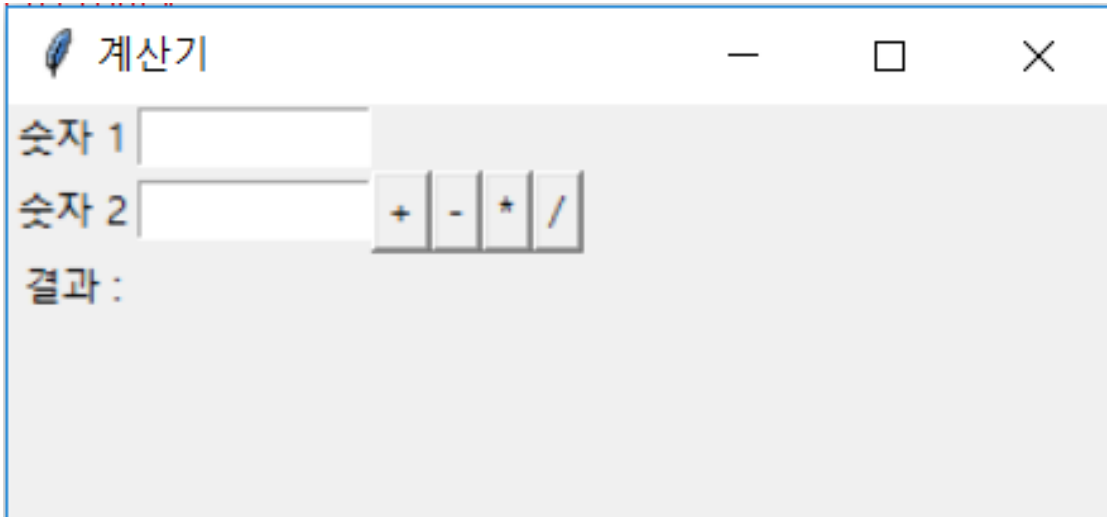
[그림 7-32] 그리드를 이용한 인터페이스 구성요소들의 배치

Tkinter를 이용한 계산기 외형 만들기

tkinter_calculator1.py

```
from tkinter import *

window = Tk()
window.title("계산기")
window.geometry('350x200')
Label(window, text = "숫자 1").grid(column = 0, row = 0)
Label(window, text = "숫자 2").grid(column = 0, row = 1)

res_label = Label(window, text = "결과 :")
res_label.grid(column = 0, row = 2)

num1 = Entry(window, width = 10)
num2 = Entry(window, width = 10)
num1.grid(column = 1, row = 0)
num2.grid(column = 1, row = 1)

btn_plus = Button(window, text = "+", command = None)
btn_minus = Button(window, text = "-", command = None)
btn_mult = Button(window, text = "*", command = None)
btn_div = Button(window, text = "/", command = None)

btn_plus.grid(column = 2, row = 1)
btn_minus.grid(column = 3, row = 1)
btn_mult.grid(column = 4, row = 1)
btn_div.grid(column = 5, row = 1)
window.mainloop()
```

- 이상의 작업이 끝나고 나면 다음과 같은 화면이 출력됨
- 아직 화면에 입력을 넣고 버튼을 눌러도 동작이 일어나지 않음



- 버튼을 누를 경우 해당 버튼에 연결된 동작을 수행하여 결과를 출력할 수 있도록 개선해 보기
- 버튼을 누르는 이벤트event가 발생하면 프로그램의 메인 루프는 발생한 이벤트가 있는지 살펴보다가 이를 확인한 뒤에 이 이벤트를 처리할 처리기 함수event handler를 부름
- 이를 콜백callback이라고도 한다.

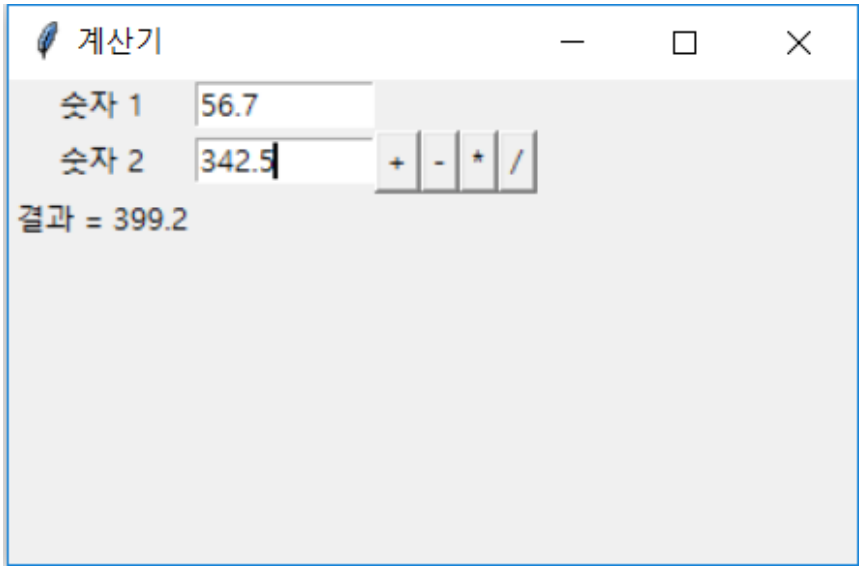
새로운 용어들
- 사용자가 마우스나 키보드를 통해 컴퓨터에 명령을 입력할 수 있다. 이러한 행위를 이벤트라고 한다
- 컴퓨터는 이 이벤트를 받아서 처리하는 이벤트 처리기 구조를 가진다. 이 이벤트 처리 함수를 콜백 함수라고 한다

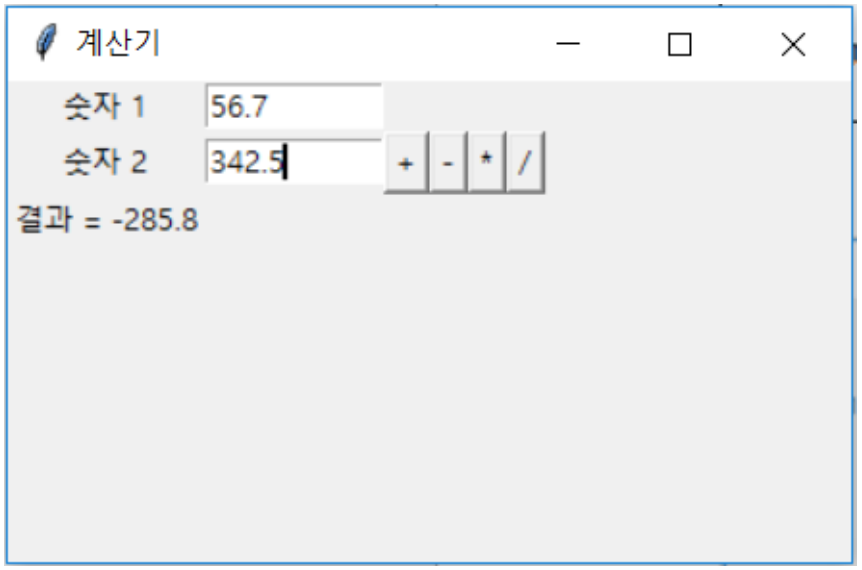
Tkinter를 이용한 간단한 계산기 만들기의 완성

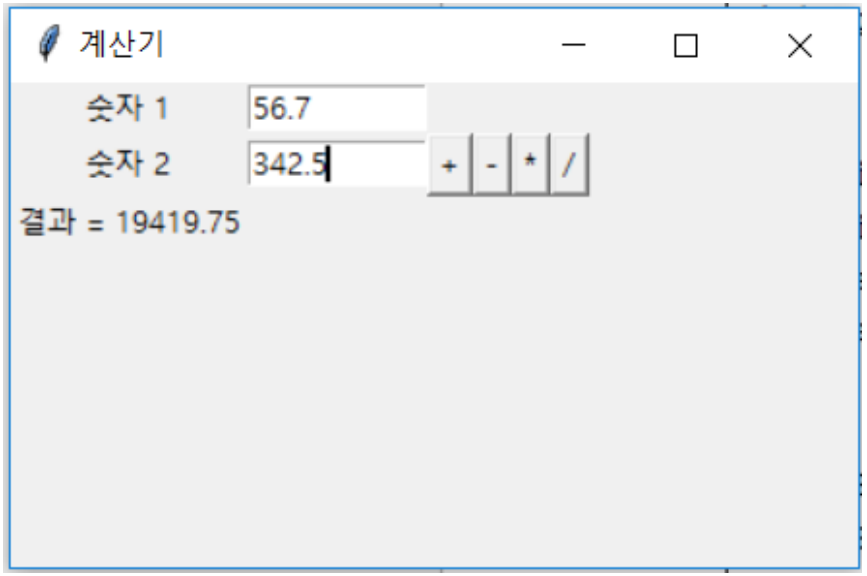
tkinter_calculator2.py

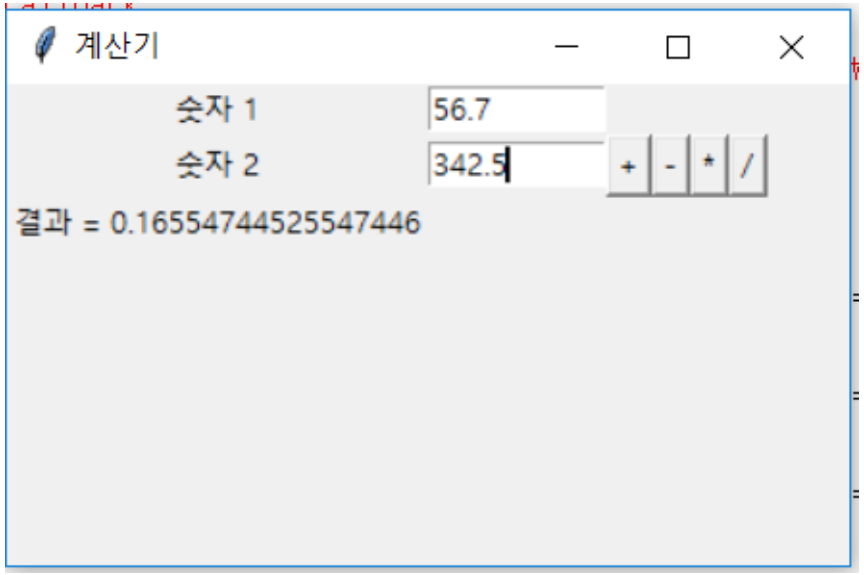
```
from tkinter import *

window = Tk()
window.title("계산기")
window.geometry('350x200')

Label(window, text = "숫자 1").grid(column = 0, row = 0)
Label(window, text = "숫자 2").grid(column = 0, row = 1)
res_label = Label(window, text = "결과 :")
res_label.grid(column = 0, row = 2)
num1 = Entry(window, width = 10)
num2 = Entry(window, width = 10)
num1.grid(column = 1, row = 0)
num2.grid(column = 1, row = 1)

def add():
    res_text = "결과 = " + str(float(num1.get()) + float(num2.get()))
    res_label.configure(text = res_text)

def subtract():
    res_text = "결과 = " + str(float(num1.get()) - float(num2.get()))
    res_label.configure(text = res_text)
def multiplication():
    res_text = "결과 = " + str(float(num1.get()) * float(num2.get()))
    res_label.configure(text = res_text)
def division():
    res_text ="결과 = " + str( float(num1.get()) / float(num2.get()))
    res_label.configure(text = res_text)

btn_plus = Button(window, text = "+", command = add)
btn_minus = Button(window, text = "-", command = subtract)
btn_mult = Button(window, text = "*", command = multiplication)
btn_div = Button(window, text = "/", command = division)
btn_plus.grid(column = 2, row = 1)
btn_minus.grid(column = 3, row = 1)
btn_mult.grid(column = 4, row = 1)
btn_div.grid(column = 5, row = 1)
window.mainloop()
```

간단한 사칙연산 계산기의 구현 결과





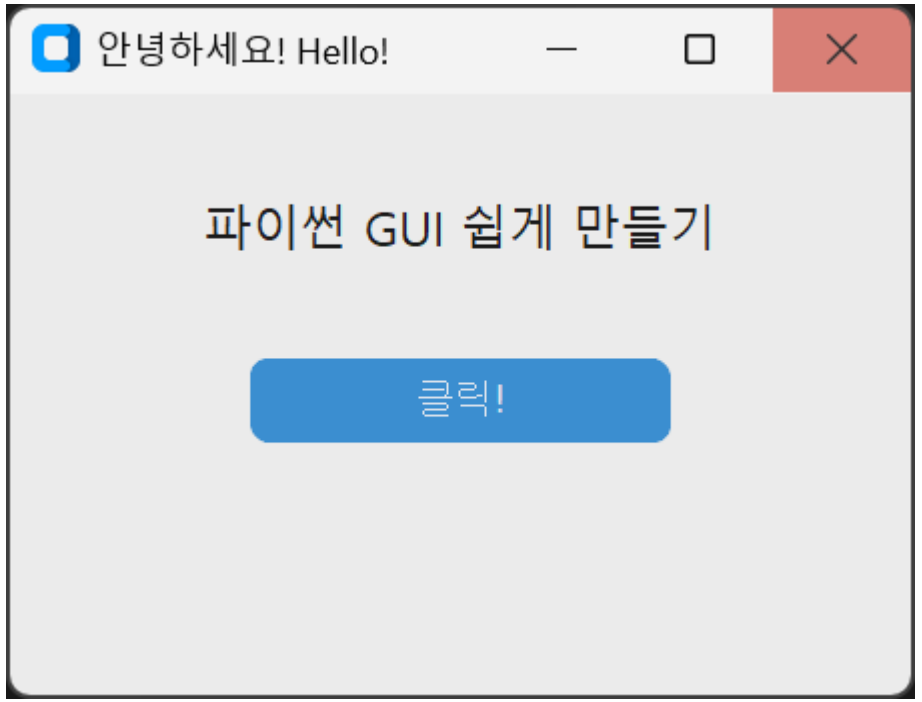




GUI 프로젝트

- Custom Tkinter: better look-and-feel
    - Pip install customtkinter

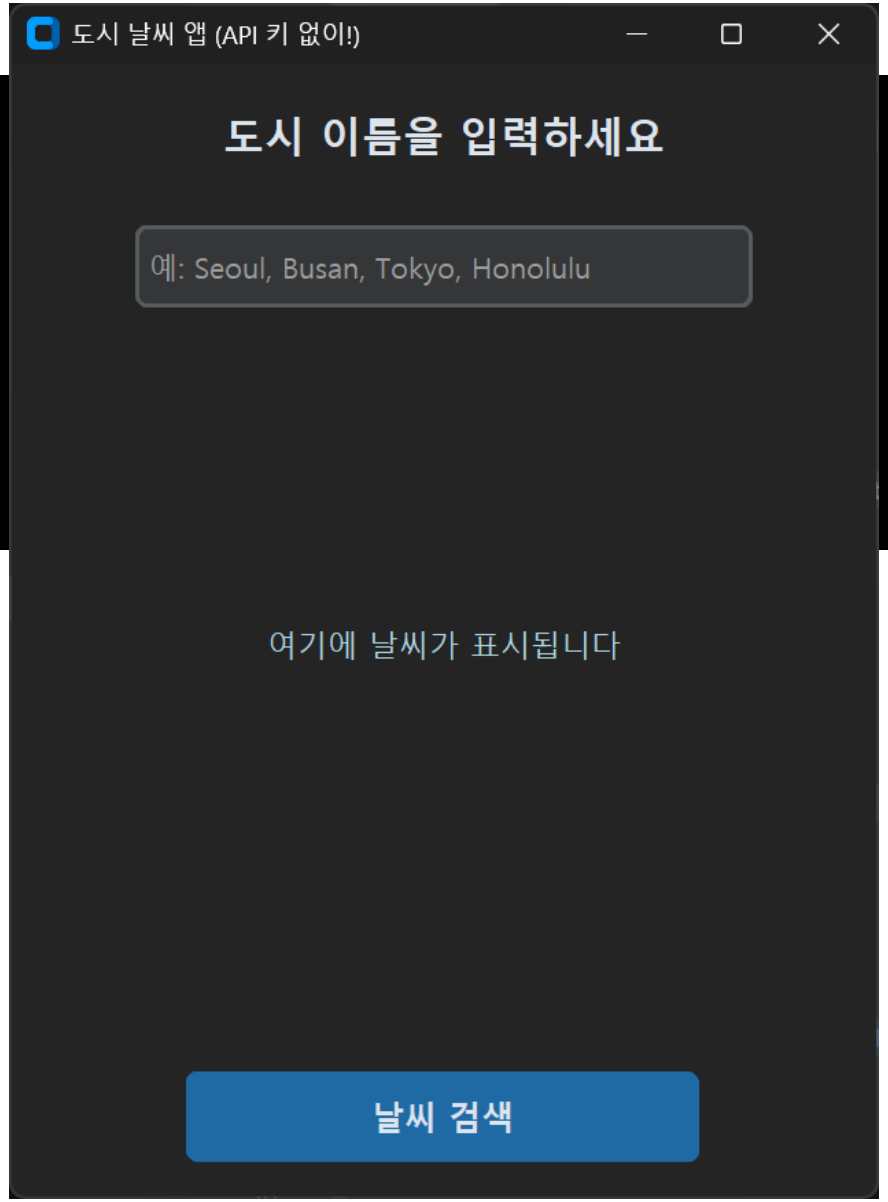
```python
import customtkinter as ctk


def change_label_text():
    label.configure(text="멋지네요! 😊")


app = ctk.CTk()
app.geometry("300x200")
app.title("안녕하세요! Hello!")


label = ctk.CTkLabel(app, text="파이썬 GUI 쉽게 만들기",
font=("맑은 고딕", 16))
label.pack(pady=30)


button = ctk.CTkButton(app, text="클릭!",
command=change_label_text)
button.pack()


app.mainloop()
```



```python
import customtkinter as ctk
import requests


# ===== GUI 설정 =====
ctk.set_appearance_mode("dark")
ctk.set_default_color_theme("blue")


app = ctk.CTk()
app.geometry("420x550")
app.title("도시 날씨 앱")

# 제목
title = ctk.CTkLabel(app, text="도시 이름을 입력하세요", font=("맑은 고딕", 20, "bold"))



# 입력창
city_entry = ctk.CTkEntry(
    app,
    placeholder_text="예: Seoul, Busan, Tokyo, Honolulu",
    width=300,
    height=40,
    font=("맑은 고딕", 14)
)
```

```python
# 결과 라벨
result_label = ctk.CTkLabel(
    app,
    text="여기에 날씨가 표시됩니다",
    font=("맑은 고딕", 15),
    wraplength=380,
    justify="center",
    text_color="lightblue"
)


# 로딩 메시지
loading_label = ctk.CTkLabel(app, text="", font=("맑은 고딕", 12), text_color="yellow")

# 검색 버튼
search_button = ctk.CTkButton(
    app,
    text="날씨 검색",
    width=250,
    height=45,
    font=("맑은 고딕", 16, "bold"),
    command=None
)
```

```python
title.pack(pady=20)
city_entry.pack(pady=10)
result_label.pack(pady=25, expand=True)
loading_label.pack(pady=5)
search_button.pack(pady=15)


# ===== 앱 실행 =====
app.mainloop()
```



```python
# ===== 날씨 가져오는 함수 (wttr.in 사용) =====
def get_weather():
    city = city_entry.get().strip()
    if not city:
        result_label.configure(text="도시 이름을 입력해주세요!", text_color="red")
        return

    loading_label.configure(text="날씨 불러오는 중...")
    app.update()

    # wttr.in 사용: 간단한 텍스트 형식
    url = f"https://wttr.in/{city}?format=%l:+%c+%t+%w+%p+%m"
    # %l: 위치, %c: 날씨 아이콘, %t: 온도, %w: 바람, %p: 강수량, %m: 달
    response = requests.get(url, timeout=10)
    if response.status_code == 200:
        weather_text = response.text.strip()
        # 예쁘게 포맷팅
        result_label.configure(
            text=f"🌍 {weather_text}",
            text_color="white",
            font=("맑은 고딕", 16)
        )
    else:
        result_label.configure(text="도시를 찾을 수 없어요", text_color="red")

    loading_label.configure(text="날씨 부르기 완료")
```

```python
# 검색 버튼
search_button = ctk.CTkButton(
    app,
    text="날씨 검색",
    width=250,
    height=45,
    font=("맑은 고딕", 16, "bold"),
    command=get_weather
)
```

파이썬 코드를 실행파일로

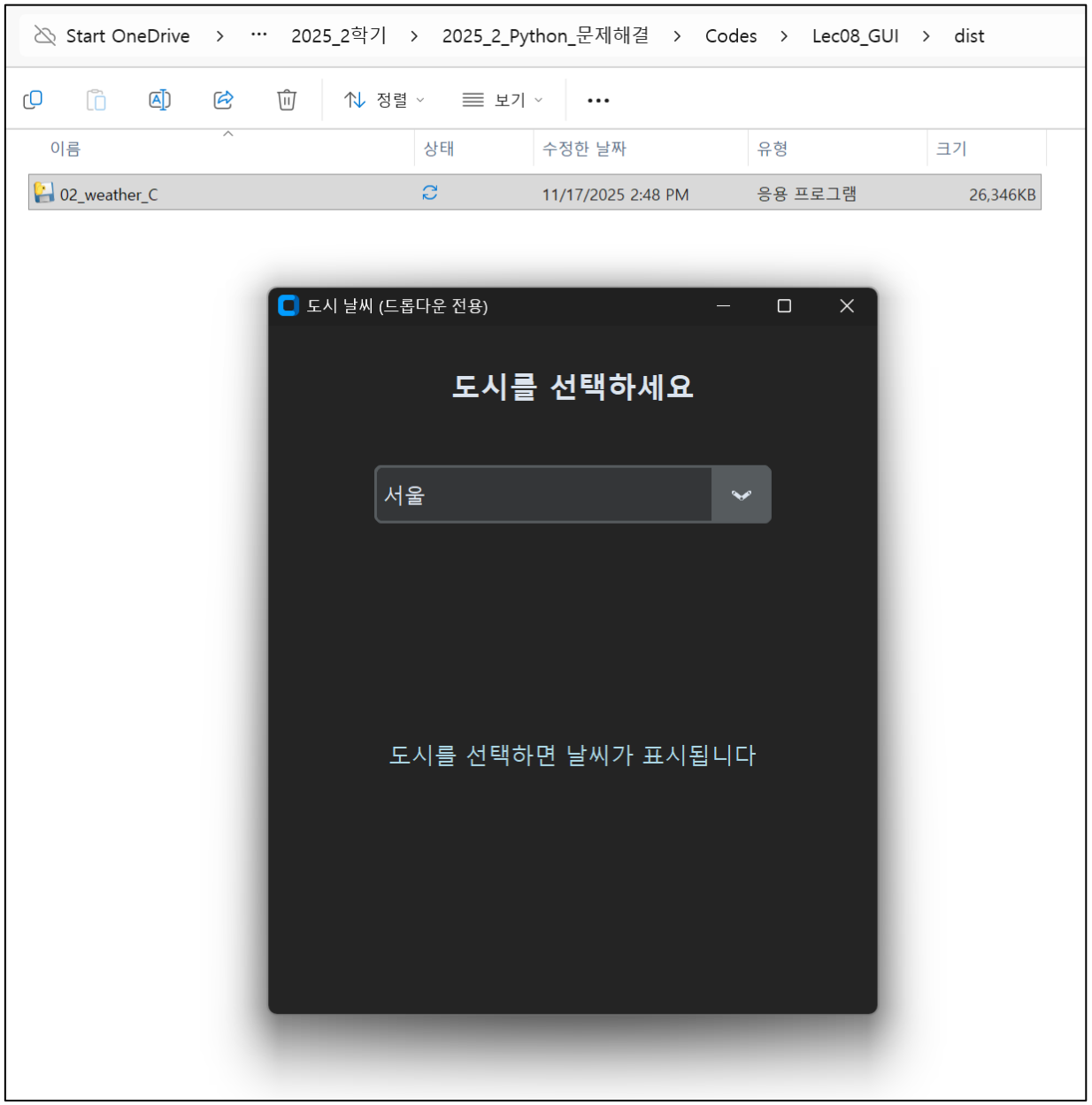
- Pyinstaller 설치
  - pip install pyinstaller

실행파일 만들기

pyinstaller --onefile --windowed .₩02_weather_C.py

```
PS D:\ymkangOneDrive\OneDrive\문서\YMKang_Work\수업_실습\2025_2학기\2025_2_Python_문제해결\Codes\Lec08_GUI> pyinstaller --onefile --windowed .\02_weather_C.py
235 INFO: PyInstaller: 6.16.0, contrib hooks: 2025.9
235 INFO: Python: 3.13.2
296 INFO: Platform: Windows-11-10.0.26100-SP0
296 INFO: Python environment: C:\Users\Administrator\AppData\Local\Programs\Python\Python313
297 INFO: wrote D:\ymkangOneDrive\OneDrive\문서\YMKang_Work\수업_실습\2025_2학기\2025_2_Python_문제해결\Codes\Lec08_GUI\02_weather_C.spec
304 INFO: Module search paths (PYTHONPATH):
['C:\\Users\\Administrator\\AppData\\Local\\Programs\\Python\\Python313\\Scripts\\pyinstaller.exe',
 'C:\\Users\\Administrator\\AppData\\Local\\Programs\\Python\\Python313\\python313.zip',
 'C:\\Users\\Administrator\\AppData\\Local\\Programs\\Python\\Python313\\DLLs',
 'C:\\Users\\Administrator\\AppData\\Local\\Programs\\Python\\Python313\\Lib',
 'C:\\Users\\Administrator\\AppData\\Local\\Programs\\Python\\Python313',
 'C:\\Users\\Administrator\\AppData\\Local\\Programs\\Python\\Python313\\Lib\\site-packages',
 'C:\\Users\\Administrator\\AppData\\Local\\Programs\\Python\\Python313\\Lib\\site-packages\\setuptools\\_vendor',
 'D:\\ymkangOneDrive\\OneDrive\\문서\\YMKang_Work\\수업_실습\\2025_2학기\\2025_2_Python_문제해결\\Codes\\Lec08_GUI']
pygame 2.6.1 (SDL 2.28.4, Python 3.13.2)
Hello from the pygame community. https://www.pygame.org/contribute.html
886 INFO: checking Analysis
887 INFO: Building Analysis because Analysis-00.toc is non existent
887 INFO: Looking for Python shared library...
887 INFO: Using Python shared library: C:\Users\Administrator\AppData\Local\Programs\Python\Python313\python313.dll
887 INFO: Running Analysis Analysis-00.toc
887 INFO: Target bytecode optimization level: 0
```

언제든 실행


Start OneDrive
2025_2학기
2025_2_Python_문제해결
Codes
Lec08_GUI
dist
정렬
보기
이름
상태
수정한 날짜
유형
크기
02_weather_C
11/17/2025 2:48 PM
응용 프로그램
26,346KB
도시 날씨 (드롭다운 전용)
도시를 선택하세요
서울
도시를 선택하면 날씨가 표시됩니다

Questions?