

9장. 윈도우 프로그래밍



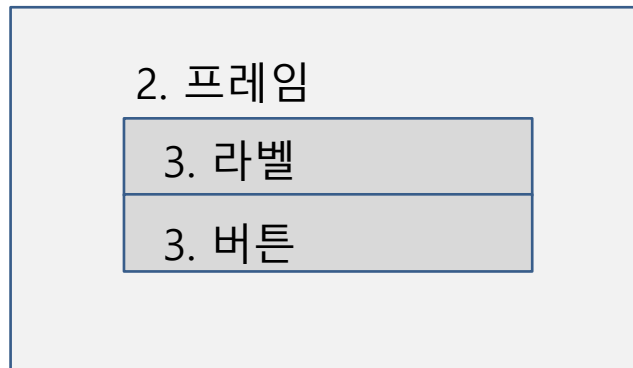
윈도우 프로그래밍

❖ GUI(Graphical User Interface)란?

그래픽 사용자 인터페이스를 줄여서 GUI라고 한다. GUI는 '화면'에 표시된 메뉴나 버튼으로 사용자와 상호 작용을 하는 간단한 프로그램이다.

tkinter 라이브러리를 사용한다. -> **import tkinter**

1. Tk 루트



개체이름	클래스
루트	Tk()
프레임	Frame
레이블	Label
입력상자	Entry
버튼	Button
출력상자	Text

윈도우 프로그래밍

■ 창(Window)

모듈 **from tkinter import ***

윈도우 생성 **root = Tk()**

윈도우 제목 **root.title()**

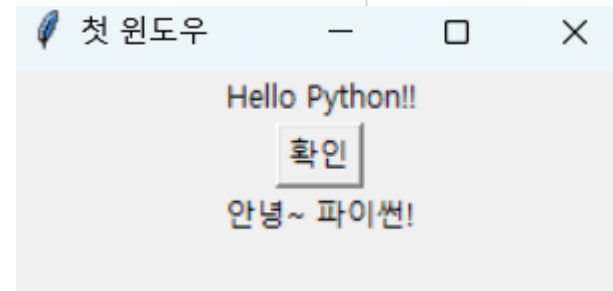
윈도우 구성 **Label, Button**

윈도우 실행 **root.mainloop()**

윈도우 프로그래밍

■ 처음 만드는 윈도우 – pack() 사용

```
def click():  
    # print("안녕~ 파이썬!")  
    demo.config(text="안녕~ 파이썬!")  
  
root = Tk()  
root.title("첫 윈도우")  
root.geometry("250x100+200+100") #너비x높이+x좌표+y좌표  
  
# 라벨과 버튼  
Label(root, text="Hello Python!!").pack()  
Button(root, text="확인", command=click).pack()  
  
# 확인 클릭 후 출력 라벨  
demo = Label(root)  
demo.pack()  
  
root.mainloop()
```



윈도우 프로그래밍

- Button(버튼) – command

Button(frame, text="확인", command=**click**).pack()

※ click에 괄호를 하면 함수 생성시점에서 작동하고, 괄호를 생략하면
클릭이 발생한 때 작동함

1. 콘솔에 출력

```
def click():  
    print("안녕~ 파이썬!")
```

2. 레이블에 출력

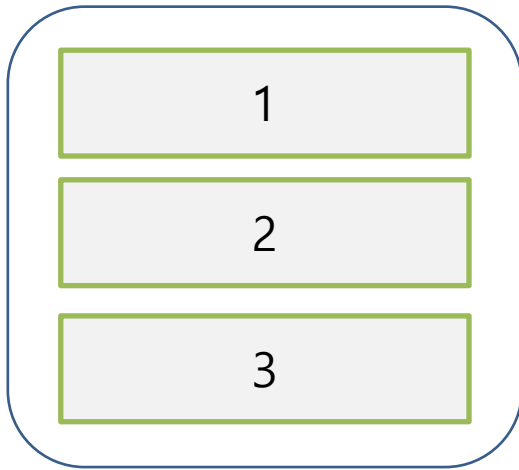
```
def click():  
    demo.config(text="안녕~ 파이썬!")
```

- 창인 크기 및 위치

root.geometry("250x100+200+100") #너비x높이+x좌표+y좌표

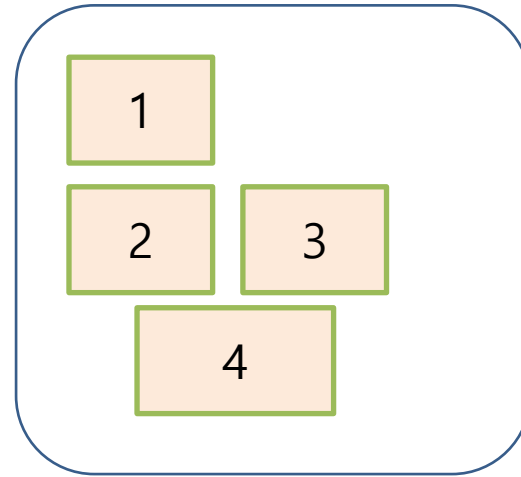
레이아웃(layout)

- 레이아웃 - pack() & grid()



pack()

하나의 컨트롤이 한 줄을 차지함



grid(행, 열)

한 줄에 여러 개의 컨트롤을 배치할 수 있음, 셀 병합도 가능

레이아웃(layout)

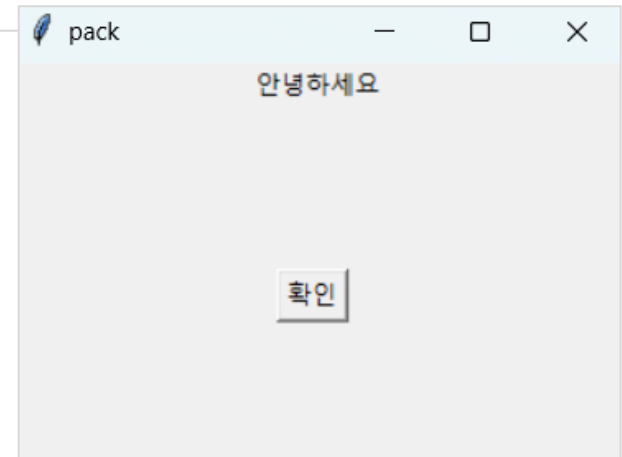
- pack() – 하나의 컨트롤이 한 줄을 차지함
- place(x, y) – 특정 좌표에 배치

```
window = Tk()
window.title('pack')
window.geometry("300x200")

Label(window, text="안녕하세요").pack()
btn = Button(window)
btn.config(text='확인')
# btn.pack()
# btn.pack(side='bottom') #left, right, top, bottom

# 특정 위치에 배치(좌표 사용)
btn.place(x = 130, y = 100)

window.mainloop()
```



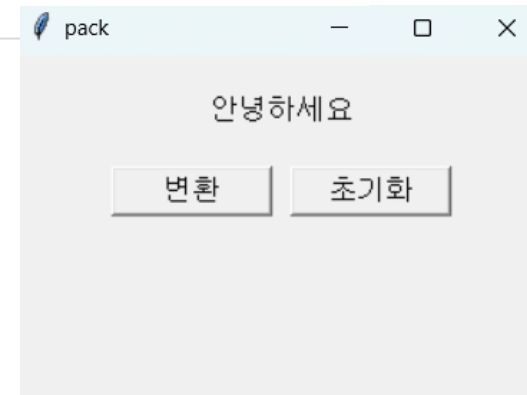
레이아웃(layout)

- `pack(padx=10)` - x축 여백 10픽셀
- `pack(pady=10)` - y축 여백 10픽셀

```
window = Tk()  
window.title('pack')  
window.geometry('300x200')
```

```
frame = Frame(window)  
frame.pack()
```

```
Label(frame, text="안녕하세요").pack(pady=20) #side="top"  
Button(frame, text="변환", width=10).pack(side="left", padx=5)  
Button(frame, text="초기화", width=10).pack(side="left", padx=5)
```

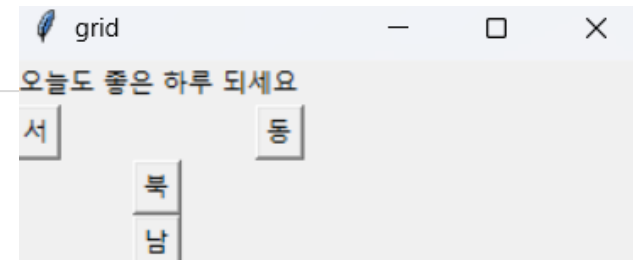


레이아웃(layout)

- grid()

```
window = Tk()  
window.title('grid')  
window.geometry("300x100")
```

```
Label(window, text="오늘도 좋은 하루 되세요").grid(row=0, column=0)  
Button(window, text='동').grid(row=1, column=0, sticky=E)  
Button(window, text='서').grid(row=1, column=0, sticky=W)  
Button(window, text='북').grid(row=2, column=0, sticky=N)  
Button(window, text='남').grid(row=3, column=0, sticky=S)  
  
window.mainloop()
```



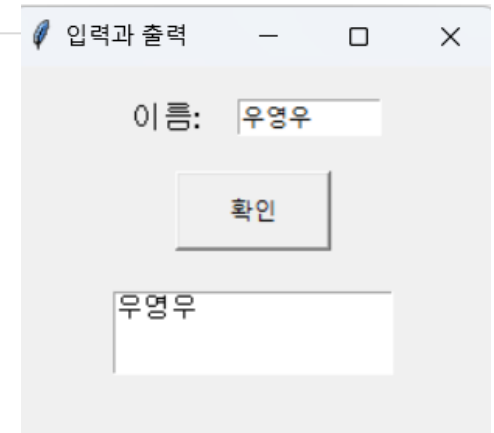
UI 프로그래밍

- 이름을 입력 받고, 화면에 출력하는 윈도우

```
root = Tk()
root.title("입력과 출력")
root.geometry("250x200+200+100")
# root.option_add("*font", "System 12") #글꼴 전체 적용

frame = Frame(root) #프레임 생성
frame.pack()         #가운데 배치

Label(frame, text="이름: ", height=3, font=('System', 12)) \
    .grid(row=0, column=0)
entry = Entry(frame, width=10) #Entry - 입력 상자 클래스
entry.grid(row=0, column=1)
Button(frame, text="확인", command=click, width=10, height=2) \
    .grid(row=1, columnspan=2)
Label(frame, text="").grid(row=2, column=0) #빈 레이블 추가
output = Text(frame, width=20, height=3) #Text - 출력상자 클래스
output.grid(row=3, columnspan=2)
```



UI 프로그래밍

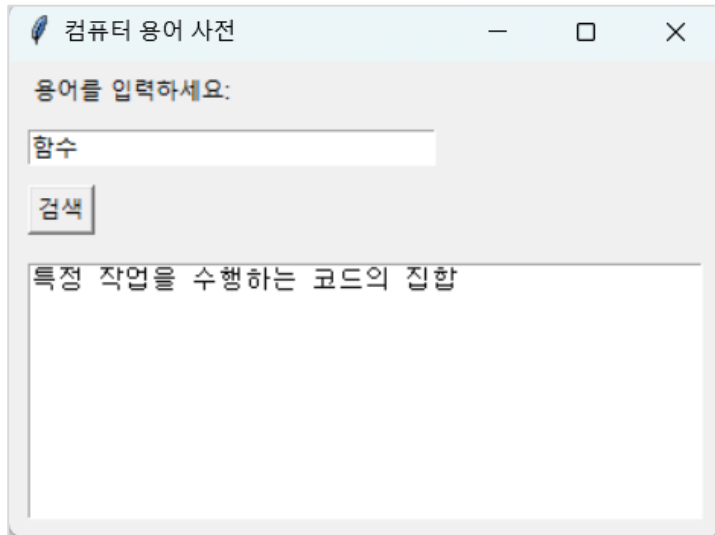
- 이름을 입력 받고, 화면에 출력하는 윈도우

```
def click():  
    text = entry.get()  
    output.delete(0.0, END)  
    output.insert(END, text)
```

#delete(): 0-1행, 0-시작문자, END-끝까지 삭제
#insert(): 끝까지 삭제 후 문자 삽입

컴퓨터 용어 사전

★ 컴퓨터 용어 사전 – dictionary 자료구조 이용



App 설명

- 용어를 미리 정의한다. – 딕셔너리 자료 구조
- 단어를 입력하고 검색 버튼을 누르면, 텍스트 상자에 뜻이 출력된다.
- 검색된 단어가 없는 경우 '사전에 없는 용어입니다..'고 출력된다.

컴퓨터 용어 사전

★ 컴퓨터 용어 사전 – dictionary.py

```
from tkinter import *

dict = {
    "알고리즘": "문제를 해결하기 위한 절차나 방법",
    "함수": "특정 작업을 수행하는 코드의 집합",
    "변수": "데이터를 저장하는 메모리 공간의 이름",
    "클래스": "객체 지향 프로그래밍에서 객체를 생성하기 위한 틀",
}

def search():
    # 입력된 단어로 사전 검색
    try:
        word = entry.get() # 입력창에서 단어 가져오기
        meaning = dict[word] # 사전에서 뜻 찾기
        output.delete(1.0, END) # 1번째 줄, 0번째 문자 = 첫 번째 위치
        output.insert(END, meaning) # 끝 위치 (텍스트의 마지막)
    except KeyError:
        output.delete(1.0, END)
        output.insert(END, "사전에 없는 용어입니다.")
```

컴퓨터 용어 사전

★ 컴퓨터 용어 사전

```
root = Tk()
root.title("컴퓨터 용어 사전")

Label(root, text="용어를 입력하세요:") \
    .grid(row=0, column=0, sticky=W, padx=10, pady=5) # 라벨 추가

entry = Entry(root, width=30)
entry.grid(row=1, column=0, sticky=W, padx=10, pady=5) # 입력창 추가


Button(root, text="검색", command=search) \
    .grid(row=2, column=0, sticky=W, padx=10, pady=5) # 버튼 추가

output = Text(root, width=50, height=10)
output.grid(row=3, column=0, sticky=W, padx=10, pady=10) # 출력창 추가



root.mainloop()
```

컴퓨터 용어 사전

■ 챗GPT 프롬프트 입력

 컴퓨터 용어 사전.txt
문서

+ 컴퓨터 용어 사전 프로그램에서 단어 등록 기능을 추가로 만들어줘

컴퓨터 용어 사전

용어 검색

이진수

검색

새 용어 등록

등록

0과 1로 이루어진 수의 체계

컴퓨터 용어 사전

■ 단어 등록 추가

```
def add_word():
    new_word = entry_new_word.get()
    new_meaning = entry_new_meaning.get()

    if new_word == "" or new_meaning == "":
        output.delete(1.0, END)
        output.insert(END, "용어와 뜻을 모두 입력하세요.")
        return

    dict[new_word] = new_meaning
    output.delete(1.0, END)
    output.insert(END, f"'{new_word}'가 사전에 등록되었습니다.")

    entry_new_word.delete(0, END)
    entry_new_meaning.delete(0, END)
```


컴퓨터 용어 사전

■ 단어 등록 추가

```
root = Tk()
root.title("컴퓨터 용어 사전")

# 검색 영역
Label(root, text="용어 검색") \
    .grid(row=0, column=0, sticky=W, padx=10, pady=5)

entry = Entry(root, width=30)
entry.grid(row=1, column=0, sticky=W, padx=10)

Button(root, text="검색", command=search) \
    .grid(row=2, column=0, sticky=W, padx=10, pady=5)
```

컴퓨터 용어 사전

■ 단어 등록 추가

```
# 등록 영역
Label(root, text="새 용어 등록") \
    .grid(row=0, column=1, sticky=W, padx=10, pady=5)

entry_new_word = Entry(root, width=30)
entry_new_word.grid(row=1, column=1, sticky=W, padx=10)
entry_new_word.insert(0, "용어")

entry_new_meaning = Entry(root, width=30)
entry_new_meaning.grid(row=2, column=1, sticky=W, padx=10)
entry_new_meaning.insert(0, "뜻")

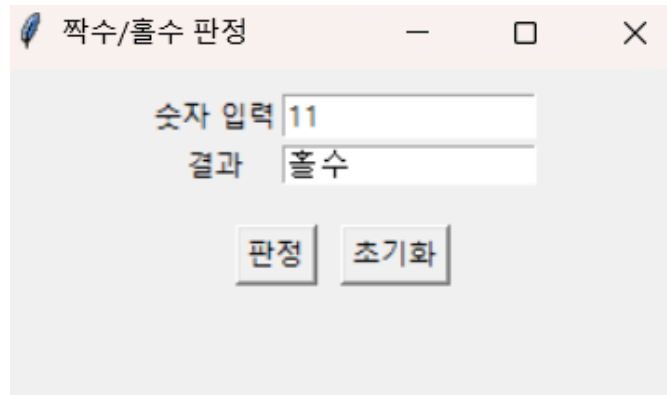
Button(root, text="등록", command=add_word) \
    .grid(row=3, column=1, sticky=W, padx=10, pady=5)

# 출력창
output = Text(root, width=70, height=10)
output.grid(row=4, column=0, columnspan=2, padx=10, pady=10)

root.mainloop()
```

짝수/홀수 판정 프로그램

- 짝수/홀수 판정



App 설명

- 숫자를 입력하면 짝수/홀수를 판정한다.
- 초기화를 수행한다.

짝수/홀수 판정 프로그램

- 짝수/홀수 판정

```
root = Tk()
root.title("짝수/홀수 판정")
root.geometry("300x150+200+200")

# 입력, 출력 프레임
io_frame = Frame(root)
io_frame.pack(pady=10) #가운데 배치

# 입력
Label(io_frame, text="숫자 입력").grid(row=0, column=0)
entry = Entry(io_frame, width=15) #입력
entry.grid(row=0, column=1)

# 출력
Label(io_frame, text="결과").grid(row=1, column=0)
result = Text(io_frame, width=15, height=1) #출력
result.grid(row=1, column=1)
```

짝수/홀수 판정 프로그램

- 짝수/홀수 판정

```
# 버튼 프레임
btn_frame = Frame(root)
btn_frame.pack(pady=5)
Button(btn_frame, text="판정", command=click).pack(side=LEFT, padx=5)
Button(btn_frame, text="초기화", command=reset).pack(side=LEFT, padx=5)

root.mainloop()
```

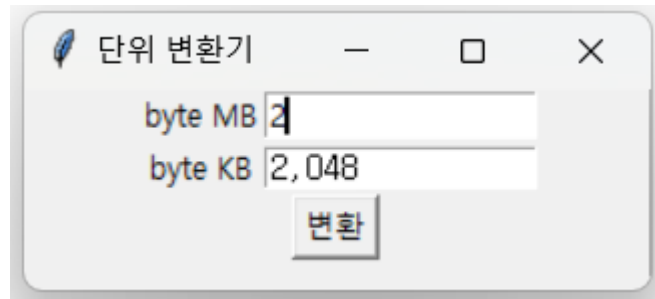
짝수/홀수 판정 프로그램

- 짝수/홀수 판정

```
def click():  
    try:  
        #숫자가 아닌 문자가 입력되면 - ValueError  
        num = int(entry.get())  
        result.delete(0.0, END) #입력된 숫자 지움  
        if num % 2 == 0:  
            result.insert(END, "짝수")  
        else:  
            result.insert(END, "홀수")  
    except ValueError:  
        #Text는 여러줄(1- 첫째줄, 0- 첫째 열)  
        result.delete(1.0, END)  
        result.insert(END, '오류')  
  
def reset():  
    entry.delete(0, END) #한 줄-0번줄  
    result.delete(1.0, END) #여러줄 - 첫째줄, 첫칼럼
```

단위 변환기 1

- 단위 변환기



App 설명

- 메모리 용량을 변환하는 단위 변환기 이다.(MB -> KB)
- 변환 명령은 함수 형태로 코딩되어 있다.

단위 변환기 1

- 단위 변환기 – 함수로 구현

```
def convert():  
    try:  
        # 입력 상자의 자료형은 문자형 -> 숫자로 변환  
        byte_mb = int(entry.get())  
        output.delete(0.0, END)      #이전 입력문자 삭제  
        byte_kb = byte_mb * 1024     #변환 상수 곱함  
        output.insert(END, byte_kb)  #변환된 kb값 저장  
    except ValueError: #숫자가 아닌 문자 입력된 경우 오류 처리  
        output.delete(0.0, END)  
        output.insert(END, "오류")
```


단위 변환기 1

- 단위 변환기 – 함수로 구현

```
window = Tk()
window.title("단위 변환기")
window.geometry("250x100+200+200")
window.option_add("*font", "돋움 13")

frame = Frame(window) #프레임 생성

Label(frame, text="byte MB").grid(row=0, column=0)
entry = Entry(frame, width=15)
entry.grid(row=0, column=1)

Label(frame, text="byte KB").grid(row=1, column=0)
output = Text(frame, width=15, height=1)
output.grid(row=1, column=1)

# 변환 버튼
Button(frame, text="변환", command=convert) \
    .grid(row=2, columnspan=2)
```

단위 변환기

■ 챗GPT 프롬프트 입력

+ 파이썬으로 단위 변환 프로그램을 만들어줘



```
===== 단위 변환 프로그램 =====
```

- 1. 길이 변환
- 2. 무게 변환
- 3. 온도 변환
- 0. 종료

메뉴 선택: 2

[무게 변환]

- 1. g → kg
- 2. kg → g

선택: 1

값 입력: 1000

1000.0 g = 1.0 kg

```
===== 단위 변환 프로그램 =====
```

- 1. 길이 변환
- 2. 무게 변환
- 3. 온도 변환
- 0. 종료

메뉴 선택: 0

프로그램 종료 🙌

단위 변환기

- 챗GPT 프롬프트 입력

단위 변환기

■ 챗GPT 프롬프트 입력

+ 위 단위 변환기를 GUI를 이용해서 만들어줘



단위 변환기

단위 변환 프로그램

값 입력: 100

cm → m

변환

결과: 1.0 m

실행 파일(.exe) 만들기

- ✓ 파일(스크립트)이 1개인 경우

```
C:\Wpyworks/gui-tkinter>pyinstaller --onefile  
--windowed 단위변환기.py
```