

Lab11. GUI Programming

CSED101 LAB



GUI Programming

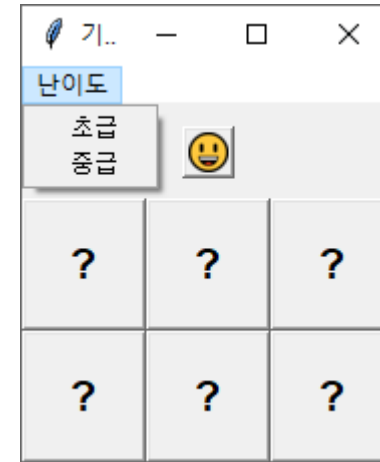
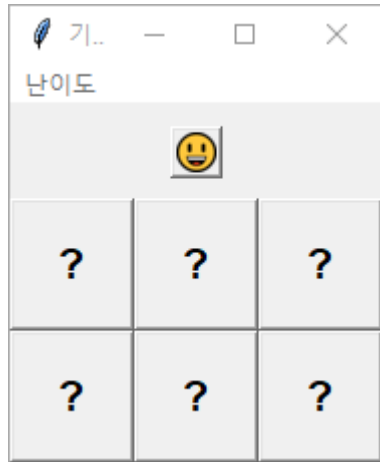
tkinter 모듈

tkinter 모듈

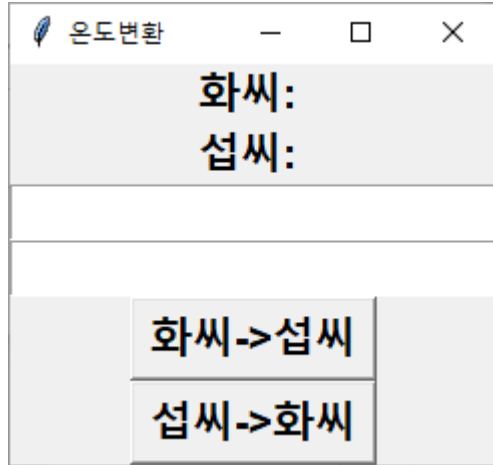
- 파이썬에서 GUI(Graphical User Interface)를 개발할 때 필요한 모듈
(파이썬 설치 시 기본으로 포함되어 설치 됨)
- 윈도우 생성하고 버튼이나 레이블과 같은 위젯을 이용하여 사용자와 상호 작용하는 프로그램을 작성할 수 있음
 - ※ 위젯: 윈도우 화면상에 나타나게 될 레이블, 버튼, 텍스트 상자 등 미리 만들어진 제어 가능한 요소
- 사각형이나 원 등의 기본적인 도형도 그릴 수 있음

실습 과제

■ 기억력 게임



기본 위젯 - 레이블, 엔트리, 버튼



```
import tkinter as tk
window = tk.Tk() # 창 생성
```

```
L1 = tk.Label(window, text="화씨: ")
L2 = tk.Label(window, text="섭씨: ")
L1.pack()
L2.pack()
```

```
E1 = tk.Entry(window)
E2 = tk.Entry(window)
E1.pack()
E2.pack()
```

```
B1 = tk.Button(window, text="화씨->섭씨")
B2 = tk.Button(window, text="섭씨->화씨")
B1.pack()
B2.pack()
```

```
window.mainloop()
```

위젯 배치 - 격자



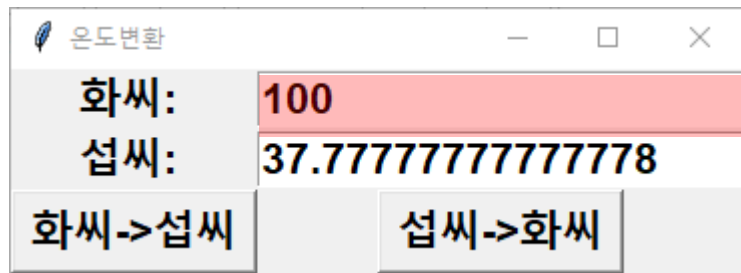
(0, 0)	(0, 1)
(1, 0)	(1, 1)
(2, 0)	(2, 1)

```
L1 = tk.Label(window, text="화씨: ")
L2 = tk.Label(window, text="섭씨: ")
L1.grid(row=0, column=0)
L2.grid(row=1, column=0)
```

```
E1 = tk.Entry(window)
E2 = tk.Entry(window)
E1.grid(row=0, column=1)
E2.grid(row=1, column=1)
```

```
B1 = tk.Button(window, text="화씨->섭씨")
B2 = tk.Button(window, text="섭씨->화씨")
B1.grid(row=2, column=0)
B2.grid(row=2, column=1)
```

버튼 이벤트 처리(1)



온도변환

화씨:	100
섭씨:	37.77777777777778
화씨->섭씨	섭씨->화씨

```
def FtoC():
```

```
    F = float(E1.get()) # 엔트리 위젯 E1을 통하여 입력 값 가져오기
```

```
    C = (F-32)*5/9
```

```
    E2.insert(0, str(C)) # 엔트리 위젯 E2의 에 삽입
```

```
    # 버튼 위젯 B1 클릭 시, FtoC함수 호출되도록 설정
```

```
B1 = tk.Button(window, text="화씨->섭씨", command=FtoC)
```

```
B2 = tk.Button(window, text="섭씨->화씨")
```

```
B1.grid(row=2, column=0)
```

```
B2.grid(row=2, column=1)
```

버튼 이벤트 처리(2)

■ 마우스 이벤트 처리

- 키보드 및 마우스를 누르는 것을 이벤트(Event)라 함
mainloop() 함수는 이러한 이벤트가 발생하기를 기다리는 함수임

마우스 작동	관련 마우스 버튼	이벤트 코드
클릭할 때	모든 버튼 공통	<Button>
	왼쪽 버튼	<Button-1>
	가운데 버튼	<Button-2>
	오른쪽 버튼	<Button-3>

```
def FtoC(event):  
    F = float(E1.get())  
    C = (F-32)*5/9  
    E2.insert(0, str(C))
```

```
B1 = tk.Button(window, text="화씨->섭씨")  
B1.bind("<Button-1>", FtoC )
```

마우스 이벤트 처리 형식

```
def 이벤트처리함수(event):  
    pass
```

```
위젯.bind("이벤트 코드", 이벤트처리함수)
```

```
def FtoC():  
    F = float(E1.get())  
    C = (F-32)*5/9  
    E2.insert(0, str(C))
```

```
B1 = tk.Button(window, text="화씨->섭씨")  
B1.bind("<Button-1>", lambda e : FtoC() )
```


레이아웃 - 프레임 사용



} Header

} Body

0. 기본창 설정

```
window = tk.Tk()
window.title("기억력게임")
```

1. Header 프레임: 초기화 버튼

```
header_frame = tk.Frame(window)
header_frame.pack()
```

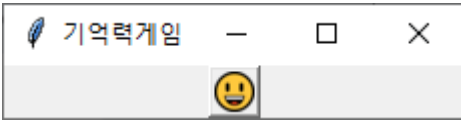
2. Body 프레임: 게임 보드(n행 x m열 개수의 버튼으로 구성)

```
body_frame = tk.Frame(window)
body_frame.pack()
```

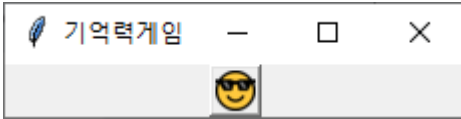
3. 메뉴 바 구성: 난이도(초급, 중급)

상단 버튼 - 이미지 출력

- 초기, 마우스 왼쪽 버튼 클릭 시



- 마우스 오른쪽 버튼 클릭 시



1. Header 프레임: 초기화 버튼에 이미지 삽입

```
header_frame = tk.Frame(window)
header_frame.pack()
```

```
img1 = tk.PhotoImage(file="imgs/smile.png") # 이미지 로드
img2 = tk.PhotoImage(file="imgs/sunglasses.png")
```

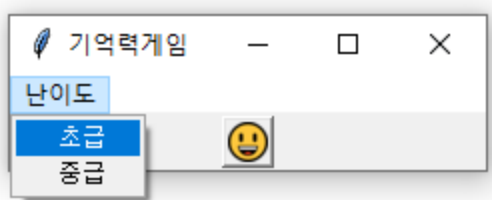
```
reset_button = tk.Button(header_frame, image=img1)
reset_button.pack() # padx=10 추가해 보기
```

```
# reset_button 을 마우스 왼쪽 버튼으로 클릭하면 img1으로 변경
#                               마우스 오른쪽 버튼으로 클릭하면 img2로 변경
```

```
reset_button.bind('<Button-1>', ??)
reset_button.bind('<Button-3>', ??)
```

※ 기존 위젯 속성 수정 시, config 메서드 사용하여 텍스트, 크기, 색상, 폰트 등을 변경할 수 있음
예시) reset_button.config(image=img2)

메뉴 바 구성



3. 메뉴 바 구성: 난이도(초급, 중급)

메뉴 바 생성

```
menu_bar = tk.Menu(window)
window.config(menu=menu_bar)
```

난이도 메뉴 생성

```
level_menu = tk.Menu(menu_bar, tearoff=0)
menu_bar.add_cascade(label="난이도", menu=level_menu)
```

난이도 메뉴에 옵션 추가

```
level_menu.add_command(label="초급", command=lambda: reset_board(2, 3)) # 2행 3열
level_menu.add_command(label="중급", command=lambda: reset_board(4, 4)) # 4행 4열
```

함수 정의

```
def reset_board(height, width):
    print(height, width)
```

게임 보드 구성(1)



게임 보드 - 함수로 구성

```
def make_board(height, width):  
    # 버튼 생성 (height x width)개 및 배치  
    i = 0  
    for r in range(height):  
        for c in range(width):  
            button=tk.Button(body_frame, width=4, height=2, text="%d" % i)  
            button.grid(row=r, column=c)  
            i += 1
```

2. Body 프레임: 게임 보드(n행 x m열 개수의 버튼으로 구성)

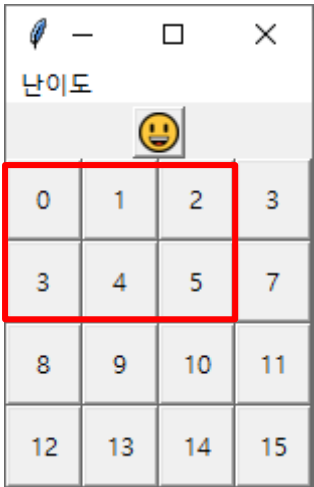
```
body_frame = tk.Frame(window)  
body_frame.pack()  
make_board(2, 3) # 초기 2행 3열 보드 생성
```

※ 난이도 테스트 하기

```
def reset_board(height, width):  
    print(height, width)  
    make_board(height, width) # 추가 후, 난이도 테스트 하기
```

게임 보드 구성(2)

- 중급 -> 초급 선택 시 기존의 버튼이 사라지지 않는 문제 발생 → 기존 버튼 삭제 필요



게임 보드 - 함수로 구성

```
def make_board(height, width):
```

기존 버튼 위젯 삭제

```
    for widget in body_frame.winfo_children():  
        widget.destroy()
```

버튼 구성 (height x width) 개

```
    i = 0
```

```
    for r in range(height):
```

```
        for c in range(width):
```

```
            button=tk.Button(body_frame, width=4, height=2, text="%d" % i)
```

```
            button.grid(row=r, column=c)
```

```
            i += 1
```

※ `winfo_children()` 메서드는 특정 위젯(예: 창, 프레임)의 모든 자식 위젯들을 리스트로 반환

게임 보드 구성(3) - 랜덤한 숫자 카드 만들어 배치

```
def make_board(height, width):
```

```
    # ... 생략 ...
```

```
    # 랜덤한 숫자 카드 만들어 배치
```

```
    pairs_left = height * width // 2 # 맞춰야 할 카드 쌍의 수
```

```
    temp_list = list(range(pairs_left)) * 2
```

```
    random.shuffle(temp_list)
```

```
    rand_list.clear()    # 2차원 리스트 전역변수
```

```
    button_list.clear() # 2차원 리스트 전역변수
```

```
    i = 0
```

```
    for r in range(height):
```

```
        row_list_rand = []
```

```
        row_list_btn = []
```

```
        for c in range(width):
```

```
            button=tk.Button(body_frame, width=4, height=2, text="%d" % temp_list[i],  
                             command=lambda row=r, col=c : on_left_click(row, col))
```

```
            button.grid(row=r, column=c)
```

```
            #button.bind("<Button-1>", lambda event, row=r, col=c : on_left_click(row, col))
```

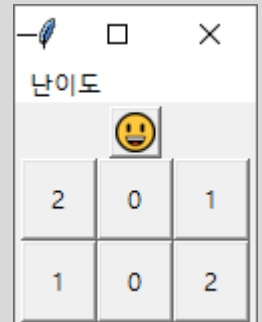
```
            row_list_rand.append(temp_list[i])
```

```
            row_list_btn.append(button)
```

```
            i += 1
```

```
        rand_list.append(row_list_rand)
```

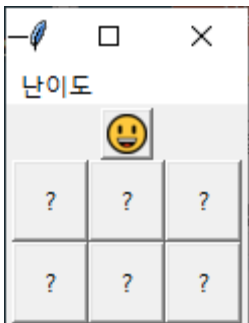
```
        button_list.append(row_list_btn)
```



```
def on_left_click(row, col):  
    print(row, col, rand_list[row][col])
```

게임 보드 구성(4)

- 2초 후, 카드 내용 숨기기



```
def make_board(height, width):  
    # ... 생략 ...  
    # 2초(2000밀리 초) 동안 카드 보여준 후, 물음표로 가리기  
    window.after(2000, hide_card_list)
```

```
def hide_card_list():  
    for row_list in button_list:  
        for button in row_list:  
            button.config(text="?")
```

버튼 이벤트 - 카드

- 아래 on_left_click() 함수를 채워 프로그램을 완성하시오.

```
def hide_card(row, col):    # 카드 물음표 출력, 버튼 활성화
    button = button_list[row][col]
    button.config(text='?' , state=tk.NORMAL, relief=tk.RAISED)

def reveal_card(row, col): # 카드 숫자 보이기, 버튼 비활성화, 누른 상태 유지
    button = button_list[row][col]
    button.config(text=rand_list[row][col], state=tk.DISABLED, relief=tk.SUNKEN)

# 전역 변수
selected_cards = []

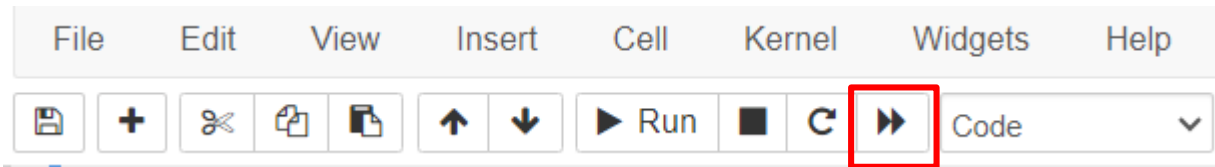
# 카드 2개 선택된 경우, 같은 숫자인지 비교
# 같은 숫자이면 누른 상태 유지, 다른 숫자이면 다시 물음표 출력(hide_card() 호출)
def on_left_click(row, col):
    pass
```

- 제출 파일명: Lab11_학번.py

■ 주피터 노트북 실행 시, 아래와 같은 에러 발생

`TclError: image "pyimage7" doesn't exist`

● 해결 방법



커널 다시 시작과 코드 전체 다시 실행