



전자피아노 작곡 프로그램

컴퓨터공학부 멀티미디어학과 20193882 홍윤아



목차



- ▶ 주제 선정 배경
- ▶ 기능 설명
- ▶ UI 설명
- ▶ 코드 설명
- ▶ 앞으로의 개선 방안



주제 선정배경

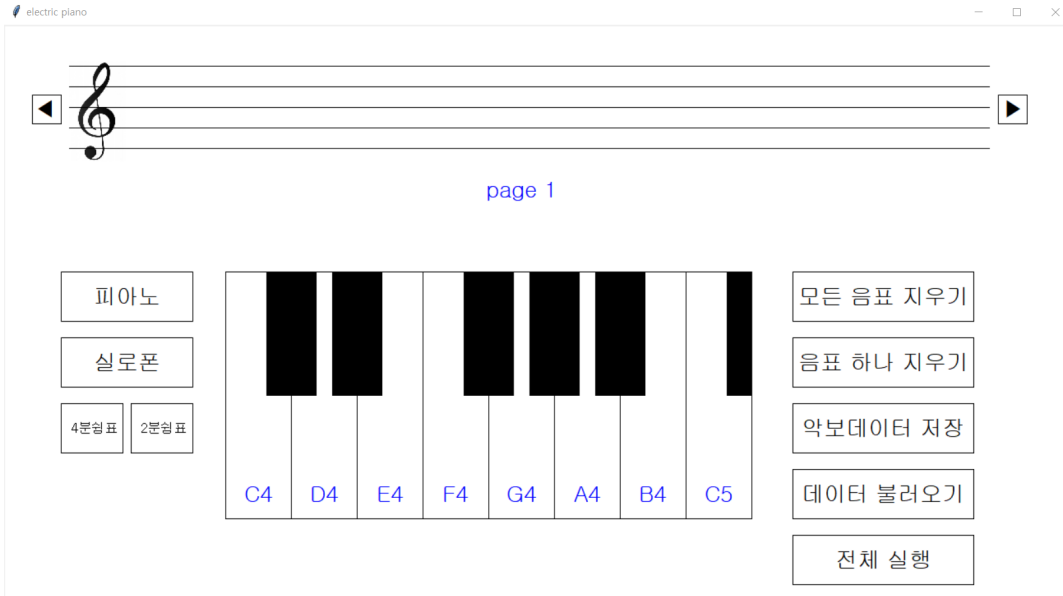
- ▶ 요즘 피아노와 피아노 음악에 관심이 있어서 전자피아노를 직접 구현하면 어떨까 싶어 이번 주제를 선정하게 되었습니다.

기능설명

1.작곡 2.악기변경 3.악보음표실행 4.악보 저장 및 수정 5.악보페이지 넘기기

- ▶ 1. 피아노 건반을 칠 때마다 악기 소리가 남
- ▶ 2.선택할 수 있는 악기 소리의 종류는 피아노와 실로폰
- ▶ 3.피아노 건반을 친 음은 악보에 그림이 그려짐
- ▶ 4.건반을 치다가 잘못된 것이 있으면 삭제가능
- ▶ 5.악보에 그려진 음표들을 모두 소리로 재생해 볼 수 있음
- ▶ 6.악보가 다 찰 때마다 다음페이지로 넘어가면서 음악이 저장됨
- ▶ 7.악보를 보듯 그동안 저장된 음표들을 페이지를 넘겨가며 볼 수 있음
- ▶ 8.현재까지 만든 음표들을 저장해서 다음에 다시 불러오고 수정할 수 있음
- ▶ 9.악보의 장이 변경될 때마다 페이지가 변경되어 표시됨

UI설명



피아노

→ 악기의 소리를 선택할 수 있다. 피아노 소리와 실로폰 소리 두 종류가 있다.

실로폰

4분침표

2분침표

→ 음악 기호 중 4분침표와 2분 침표를 악보에 추가할 수 있다.

모든 음표 지우기

→ 피아노로 친 악보에 표시된 음표들을 악보그림과 전체리스트에서 모두 삭제한다. 페이지와 음표의 위치는 1로 리셋된다.

음표 하나 지우기

→ 피아노로 친 악보에 표시된 음표들 중 가장 마지막에 저장된 음표를 악보그림과 전체 리스트에서 제거한다.

악보데이터 저장

→ 현재까지 악보에 표시된 음들의 위치리스트(npos_list)와 해당하는 피아노 음들의 리스트(sound_list)를 저장한다.

데이터 불러오기

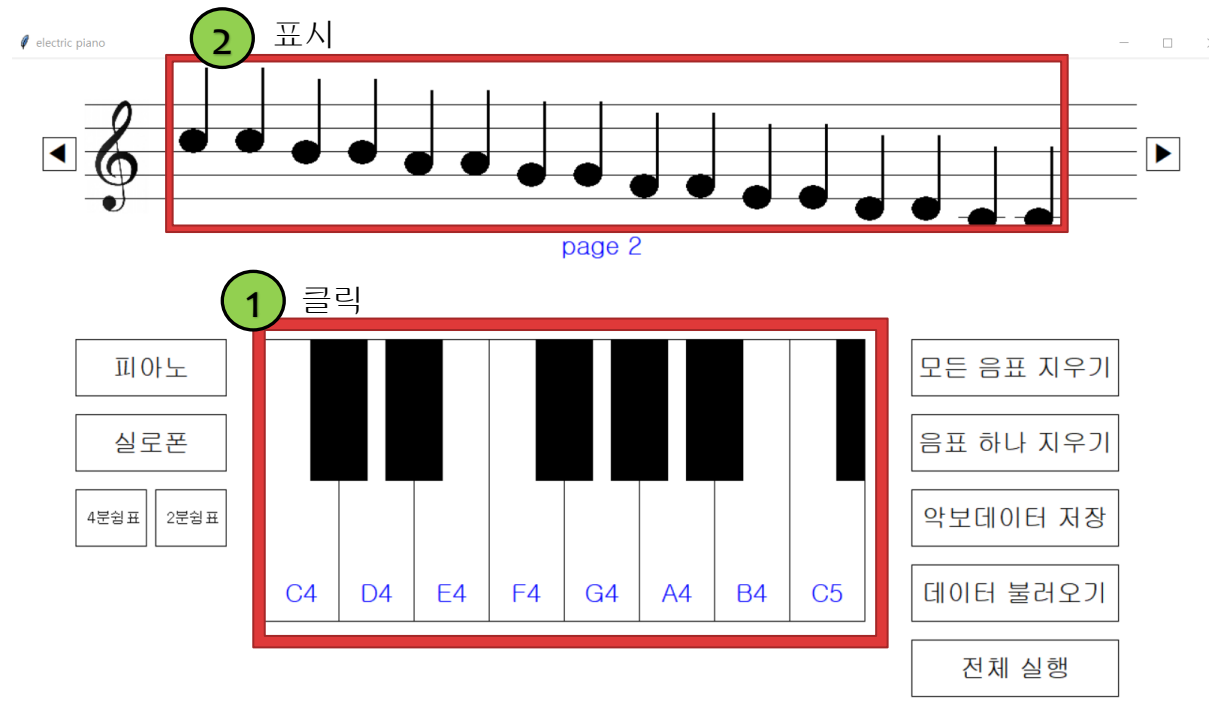
→ 앞의 악보데이터 저장 버튼을 눌러 저장된 npos_list(음표들의 위치 리스트)와 sound_list(피아노 음들 리스트)파일을 불러온다.

전체 실행

→ 악보에 표시된 모든 음표들을 실제 악기를 치는 것처럼 실행한다.

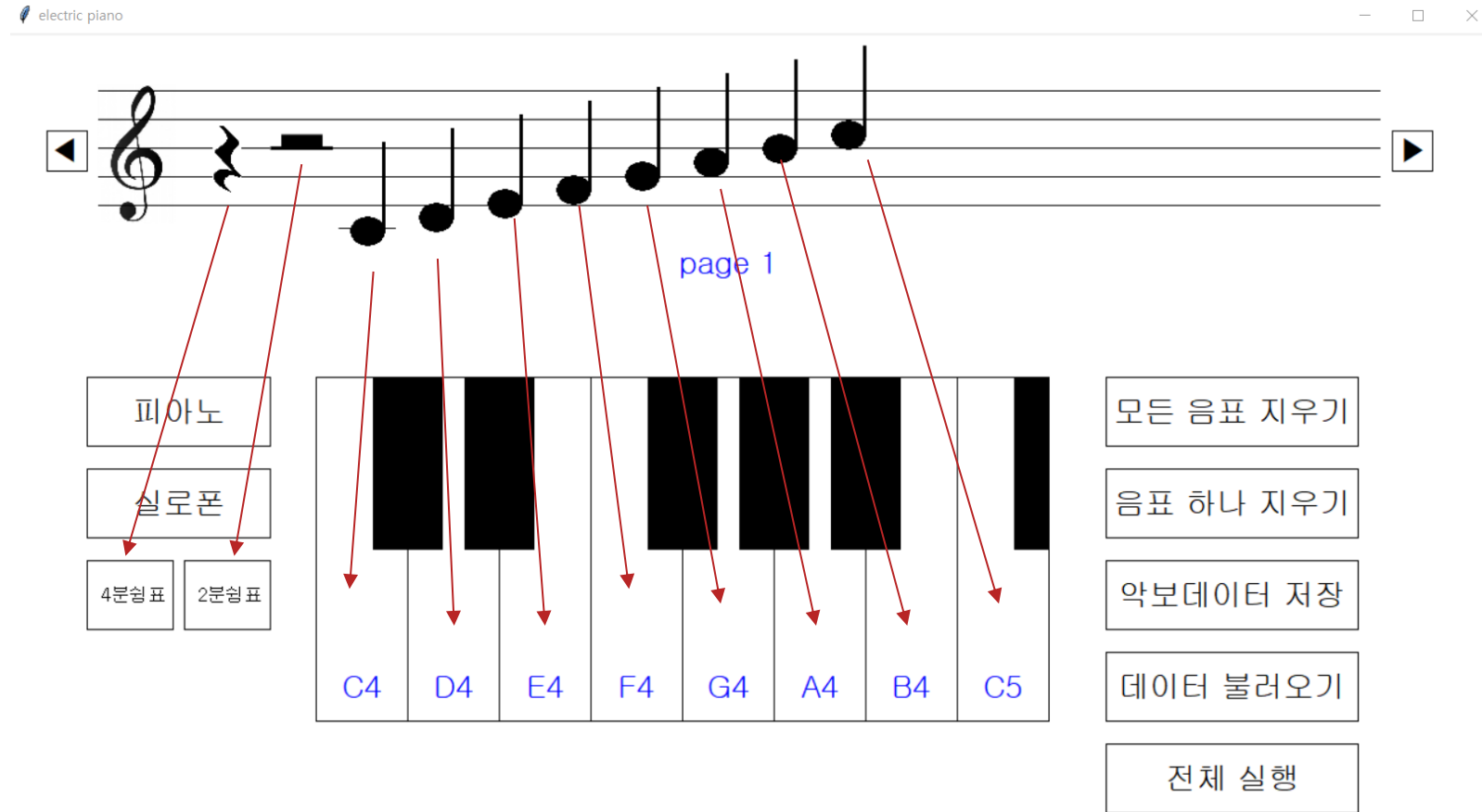
UI설명

1. 피아노 버튼을 눌렀을때
->악보에 음표 표시와 음 소리 재생



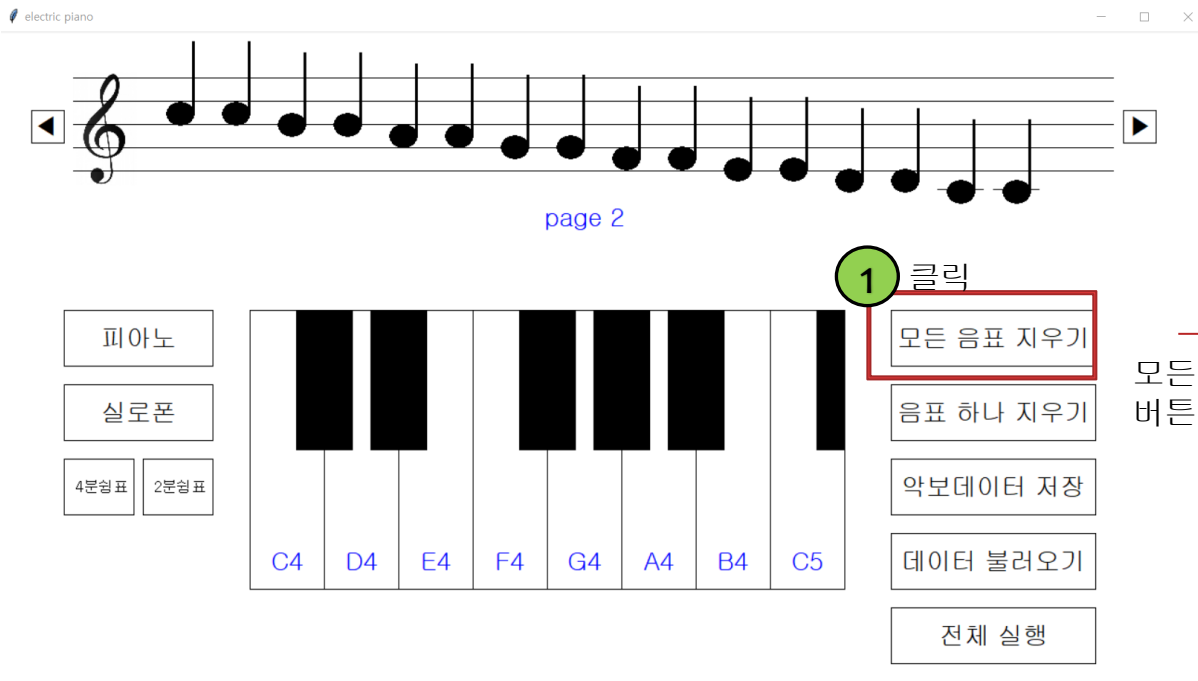
UI설명

1-1. 피아노 건반과 4분침표, 2분침표 버튼을 클릭했을 때
->음악 기호들이 모두 악보에 그려짐(소리는 영상시연 때)

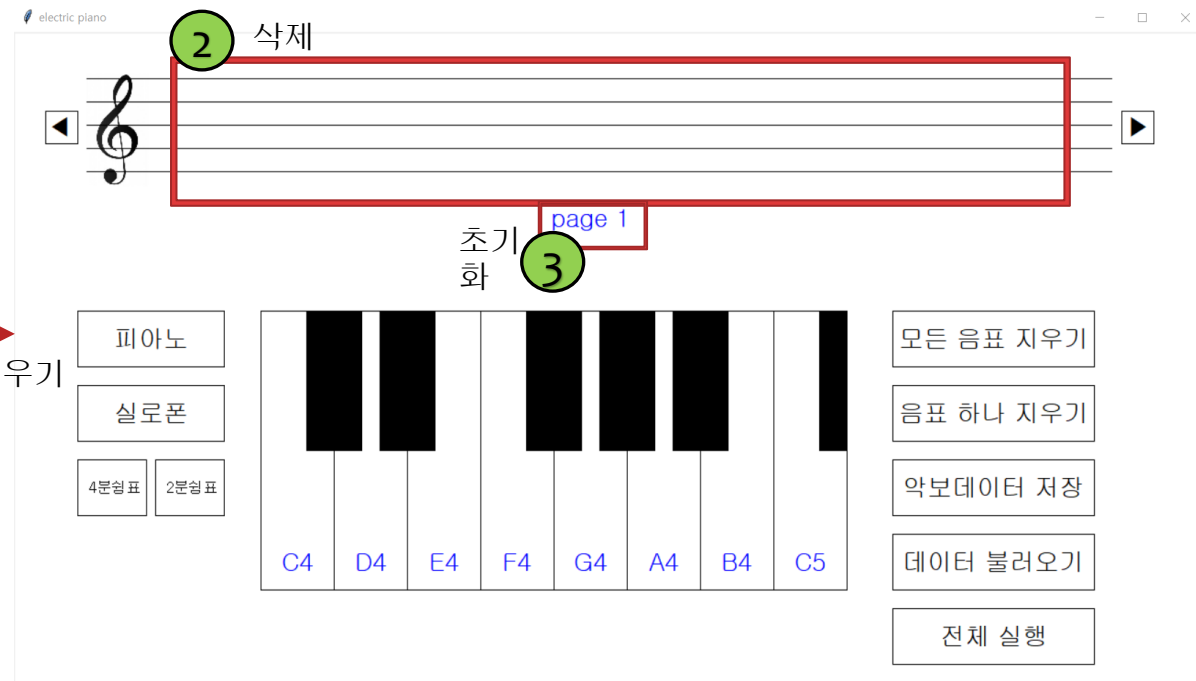


UI설명

2. 모든 음표 지우기 버튼을 눌렀을때
->악보에 표시되었던 음표 전체 삭제, 페이지 초기화



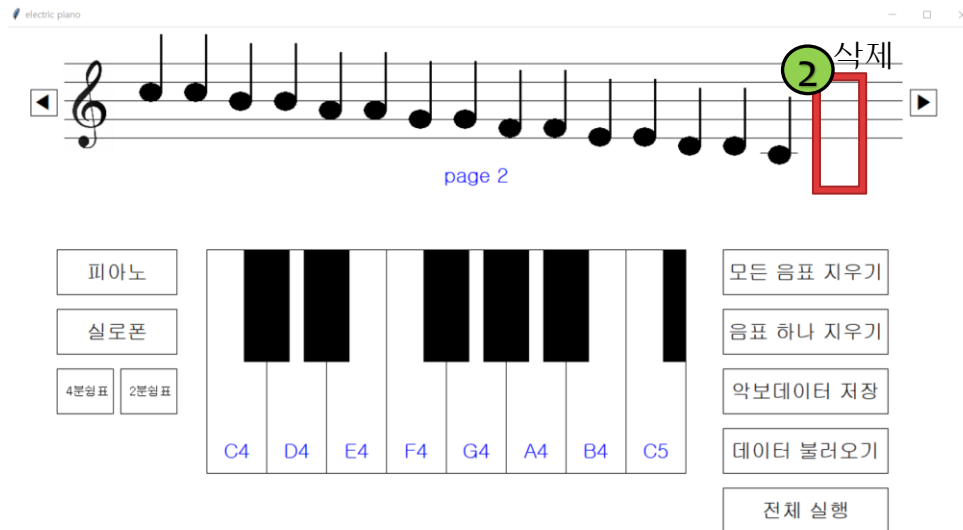
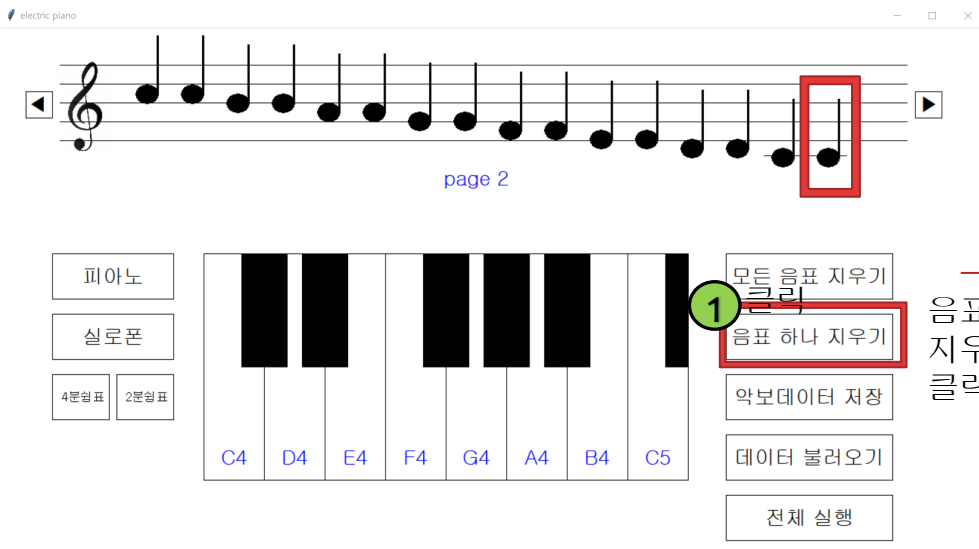
모든 음표 지우기
버튼 클릭



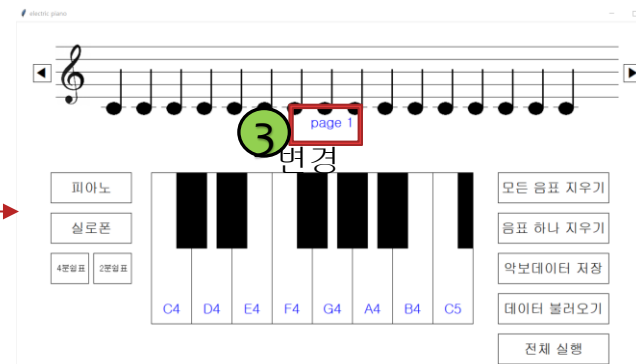
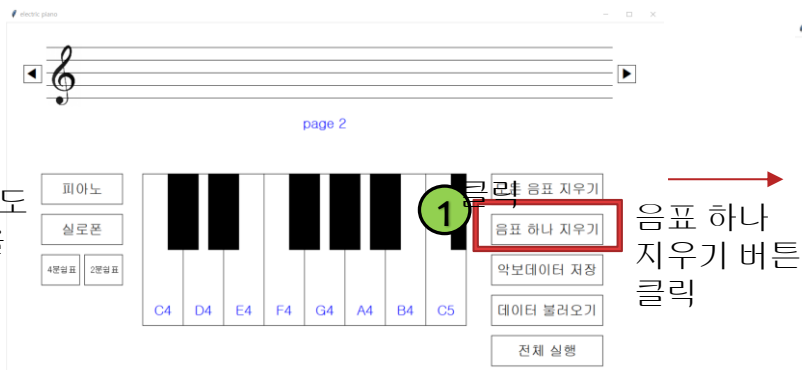
UI설명

3.음표 하나 지우기 버튼을 눌렀을때
->악보에 표시되었던 음표 하나 삭제, 페이지 초기화

1.음표가 1개 이상
그려져있을 경우



2.음표가 아무것도
그려져있지 않을 경우

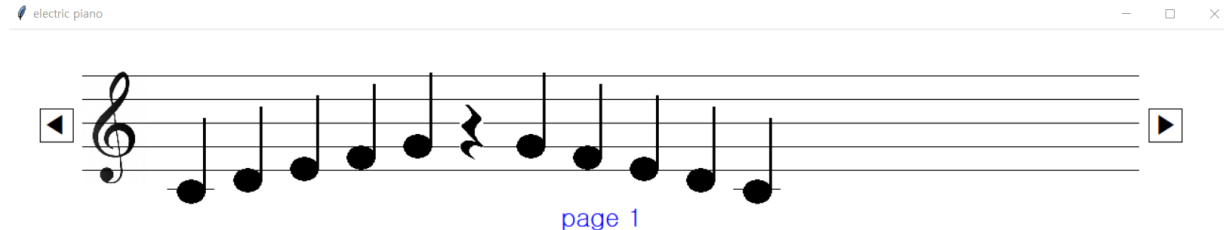


UI설명

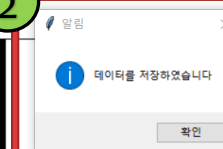
4. 악보데이터 저장버튼을 눌렀을때
->악보에 표시된 모든 음표의 위치리스트와
음표소리인덱스 리스트를 npy파일로 저장함



악보데이터 저장
버튼 클릭



알림



전

이름	수정한 날짜	유형	크기
pycache_	2022-12-02 오전 9:30	파일 폴더	
draw_UI.py	2022-12-02 오전 9:30	JetBrains PyChar...	3KB
main.py	2022-12-02 오후 12:25	JetBrains PyChar...	19KB

후



수정한 날짜	유형	크기
2022-12-02 오전 9:30	파일 폴더	
2022-12-02 오후 2:51	파일 폴더	
2022-12-02 오전 9:30	JetBrains PyChar...	3KB
2022-12-02 오후 12:25	JetBrains PyChar...	10KB

이름	수정한 날짜	유형	크기
pos_list1.npy	2022-12-02 오후 2:51	NPY 파일	1KB
sound_list1.npy	2022-12-02 오후 2:51	NPY 파일	1KB



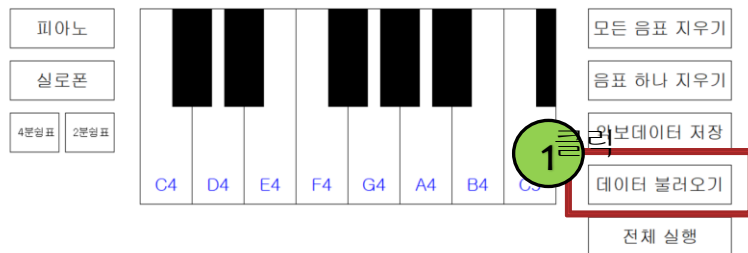
시설명

5.데이터 불러오기 버튼을 눌렀을 때
->저장된 데이터를 불러와서(pos,sound순으로
선택) 악보에 그림

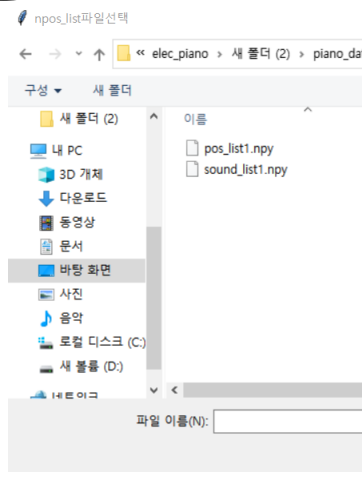


page 1

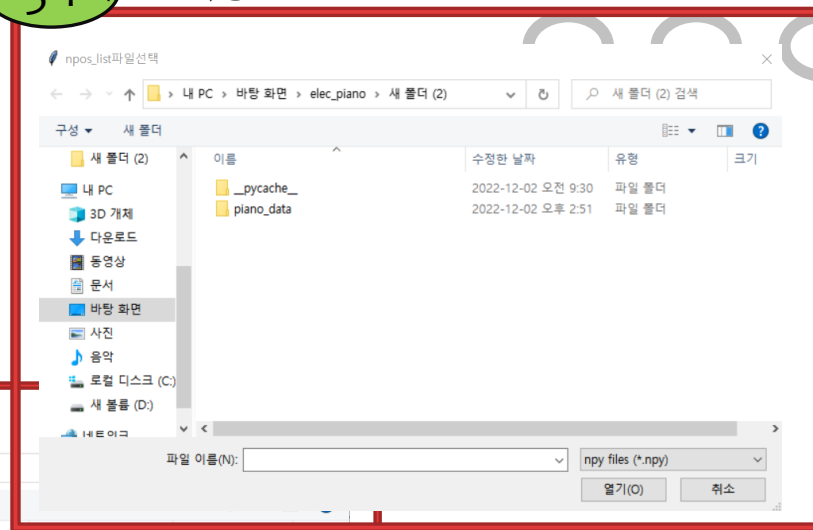
1.악보에 아무것도
없는 경우



3-2 파일 선택



3-1 파일 선택창

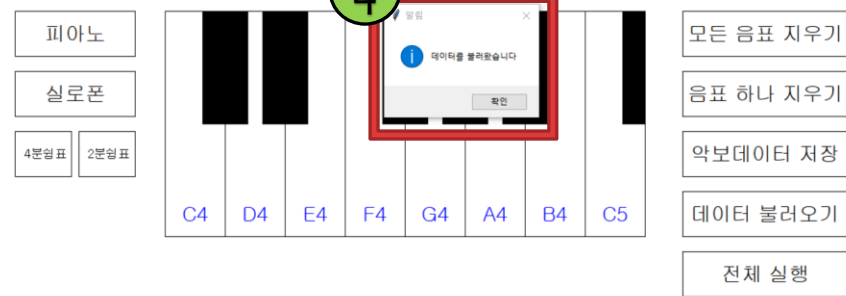


5 새로 표시됨

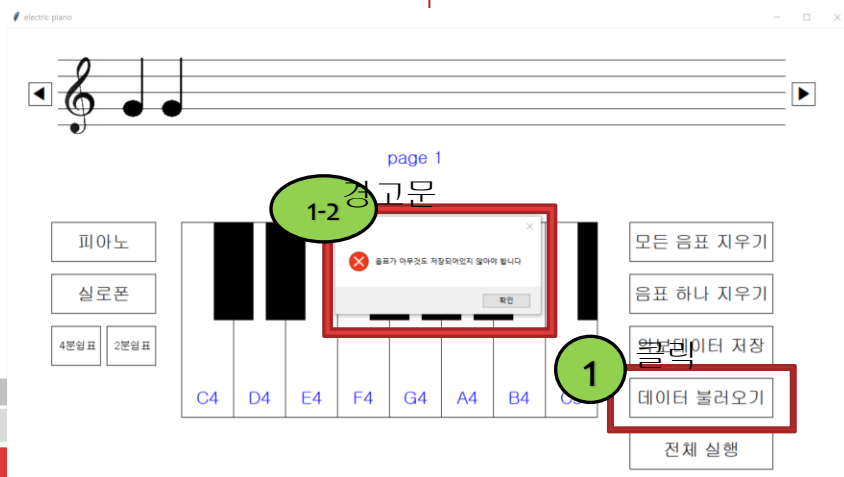


알림

4



2.악보에 음표가
그려져있는 경우

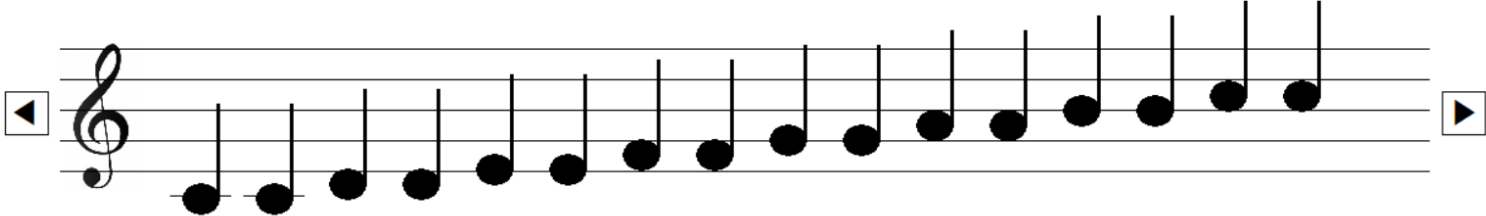


UI설명

6.전체 실행 버튼을 눌렀을 때

->악보에 그려진 음표들을 모두 실행함.(시연영상에서)

electric piano

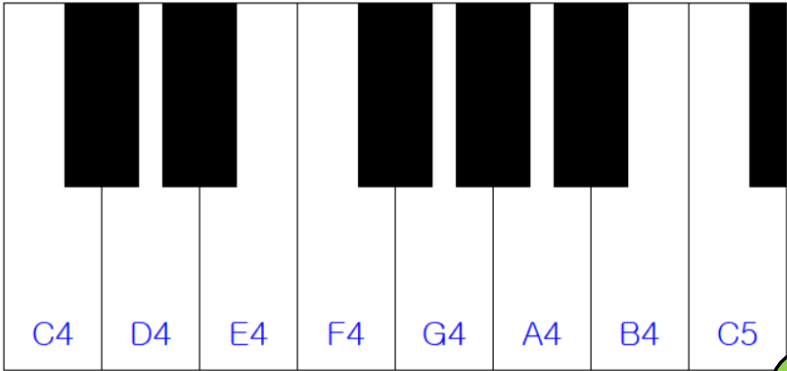


page 1

피아노

실로폰

4분침표 2분침표



모든 음표 지우기

음표 하나 지우기

악보데이터 저장

데이터 불러오기

클릭

1 전체 실행

시설명

7.악보 넘어가기 버튼을 눌렀을 때

->악보가 앞뒤로 넘어감, 페이지가 끝장, 첫장이면 넘어가지 않음

1.오른쪽 버튼 클릭

2.왼쪽 버튼 클릭

1. 클릭

2. 클릭

3. 변경

3. 변경

page 1

page 2

피아노

실로폰

4분음표 2분음표

C4 D4 E4 F4 G4 A4 B4 C5

모든 음표 지우기

음표 하나 지우기

악보데이터 저장

데이터 불러오기

전체 실행

4. 알림

page 2

알림

다음 장이 없습니다

확인

피아노

실로폰

4분음표 2분음표

C4 D4 E4 F4 G4 A4 B4 C5

모든 음표 지우기

음표 하나 지우기

악보데이터 저장

데이터 불러오기

전체 실행

4. 알림

page 1

알림

첫 번째 페이지입니다

확인

피아노

실로폰

4분음표 2분음표

C4 D4 E4 F4 G4 A4 B4 C5

모든 음표 지우기

음표 하나 지우기

악보데이터 저장

데이터 불러오기

전체 실행

코드설명 draw_UI.py



```
def draw_screen(canvas):
    for x in range(50, 170, 25): #오선지
        canvas.create_line(80, x, 1200, x, fill="black")

    for x, note in zip(range(270, 891, 80), ["C4", "D4", "E4", "F4", "G4", "A4", "B4", "C5"]): #흰건반
        canvas.create_rectangle(x, 300, x + 80, 600, fill='white', outline='black', width=1)
        canvas.create_text(x + 40, 570, text=note, font=("나눔고딕코딩", 20), fill="blue")

    for x in range(320, 721, 80): #검은건반
        if x == 480:
            continue
        canvas.create_rectangle(x, 300, x+60, 450, fill='black', outline='black', width=1)
    canvas.create_rectangle(880, 300, 880+30, 450, fill='black', outline='black', width=1)
    p1 = canvas.create_text(630, 200, text=f"page 1", font=("나눔고딕코딩", 20), fill="blue") #페이지장
    draw_buttons(canvas)
    return p1
```

```
def draw_buttons(canvas):
    canvas.create_rectangle(960, 300, 1180, 360, fill='white', outline='black', width=1)
    canvas.create_text(1070, 330, text="모든 음표 지우기", font=("나눔고딕코딩", 20), fill="black")
    canvas.create_rectangle(960, 380, 1180, 440, fill='white', outline='black', width=1)
    canvas.create_text(1070, 410, text="음표 하나 지우기", font=("나눔고딕코딩", 20), fill="black")
    canvas.create_rectangle(960, 460, 1180, 520, fill='white', outline='black', width=1)
    canvas.create_text(1070, 490, text="악보데이터 저장", font=("나눔고딕코딩", 20), fill="black")
    canvas.create_rectangle(960, 540, 1180, 600, fill='white', outline='black', width=1)
    canvas.create_text(1070, 570, text="데이터 불러오기", font=("나눔고딕코딩", 20), fill="black")
    canvas.create_rectangle(960, 620, 1180, 680, fill='white', outline='black', width=1)
    canvas.create_text(1070, 650, text="전체 실행", font=("나눔고딕코딩", 20), fill="black")
```

```
canvas.create_rectangle(70, 300, 230, 360, fill='white', outline='black', width=1)
canvas.create_text(150, 330, text="피아노", font=("나눔고딕코딩", 20), fill="black")
canvas.create_rectangle(70, 380, 230, 440, fill='white', outline='black', width=1)
canvas.create_text(150, 410, text="실로폰", font=("나눔고딕코딩", 20), fill="black")
canvas.create_rectangle(70, 460, 230, 520, fill='white', outline='black', width=1)
canvas.create_text(110, 490, text="4분쉼표", font=("나눔고딕코딩", 12), fill="black")
canvas.create_rectangle(155, 460, 230, 520, fill='white', outline='black', width=1)
canvas.create_text(195, 490, text="2분쉼표", font=("나눔고딕코딩", 12), fill="black")
```

```
canvas.create_rectangle(35, 85, 70, 120, fill='white', outline='black', width=1)
canvas.create_text(50, 102, text="◀", font=("나눔고딕코딩", 20), fill="black")
canvas.create_rectangle(1210, 85, 1245, 120, fill='white', outline='black', width=1)
canvas.create_text(1228, 102, text="▶", font=("나눔고딕코딩", 20), fill="black")
```

- ▶ UI를 만드는 파일
- ▶ draw_screen은 오선지와 피아노를 그림.
- ▶ draw_buttons는 여러 기능들을 실행할 버튼들을 그림.

코드설명 main.py

```
def main():
    global canvas, page, p1
    width = 1300; height = 700
    app = Tk()
    app.title("electric piano")
    canvas = Canvas(app, width=width, height=height, bg="white")
    canvas.pack(fill=BOTH, expand=True)
    photoImage = PhotoImage(file="C:/Users/USER/Desktop/elec_piano/high.png").subsample(3) #draw high note
    canvas.create_image(80, 45, anchor=NW, image=photoImage)

    global rest_4, rest_2
    rest_4 = PhotoImage(file="C:/Users/USER/Desktop/elec_piano/image01.png").subsample(3) #4분쉼표
    rest_2 = PhotoImage(file="C:/Users/USER/Desktop/elec_piano/image02.png").subsample(10) #2분쉼표
    p1 = draw_screen(canvas)
    app.bind("<Button-1>", callback_mouse)
    app.mainloop()
```

- ▶ 맨 처음 실행하는 함수
- ▶ 화면 크기는 가로 1300, 세로 700으로 지정
- ▶ Tkinter과 canvas 지정
- ▶ 화면의 높은 음 자리표와 4분쉼표, 2분쉼표를 tkinter의 PhotoImage로 지정
- ▶ Tkinter의 canvas화면을 왼쪽 마우스로 누를 때 마다 callback_mouse함수가 실행

코드설명 main.py

```
##한 악보당 16개
SOUND_FOLDER = r"C:\Users\USER\Desktop\elec_piano\soundfiles"
notes = {"do-c4.wav":0, "re-d4.wav":1, "mi-e4.wav":2, "fa-f4.wav":3, "sol-g4.wav":4, "la-
a4.wav":5, "si-b4.wav":6, "do-c5.wav":7,
        "xylophone-c1.wav":8, "xylophone-d1.wav":9, "xylophone-e1.wav":10, "xylophone-
f1.wav":11, "xylophone-g1.wav":12,
        "xylophone-a1.wav":13, "xylophone-b1.wav":14, "xylophone-c2.wav":15, "rest_4":16,
"rest_2":17}
instrument = 0
note_pos = 1 #음표 위치
note_gap = 60 #음표간 간격
start_pos = 120 #첫번째 음표 위치
last_pos = start_pos + 30 #음표크기지정
notes_list = [] #그려진 음표 객체 리스트
npos_list = [] #음표를 그리는 위치 좌표 저장리스트
sound_list = [] #전체 음표 음실행 리스트
page = 1 #전체 페이지
now_p = 1 #현재 페이지
```

#모든 음표 지우기

```
def remove_all():
    global canvas, note_pos, now_p, p1, page, sound_list
    for a in notes_list:
        if isinstance(a, list):
            for i in a:
                canvas.delete(i) # note
            canvas.delete(a)
    notes_list.clear()
    npos_list.clear()
    sound_list.clear()
    note_pos = page = now_p = 1 #음표위치, 전체페이지, 현재페이지 초기화
    canvas.delete(p1) #페이지 글씨객체 삭제
    p1 = canvas.create_text(630, 200, text=f"page {now_p}",
font=("나눔고딕코딩", 20), fill="blue")
```

- ▶ 프로젝트에 전반적으로 사용되었던 변수들
- ▶ notes딕셔너리는 데이터를 불러올 때 사용
- ▶ remove_all함수는 모든 음표를 지울때 사용
- ▶ notes_list는 악보에 그려지는 음표들의 객체를 담고 있는 리스트
- ▶ 음표들은 타원과 직선이 결합되어 있으므로 리스트에 저장되어 있고 이를 삭제하여 음표를 악보에서 삭제
- ▶ 모든 음표를 지우므로 앞으로 그려질 음표의 위치와 전체 페이지와 현재 페이지를 모두 1로 초기화



코드설명 main.py



#마지막 음표 하나 지우기

```
def remove_one():
    global note_pos, now_p, p1, page, sound_list
    if note_pos == 1: #마지막 장에 음표가 하나 있을때
        if now_p <= 1:
            showwarning("경고", "첫 페이지입니다")
            return
        print(now_p)
        now_p -= 1
        page -= 1
        note_pos = 16
        showinfo("알림", f"page {now_p}로 넘어갑니다.")
        for i, (a,b,c,d) in enumerate(npow_list[16*(now_p-1):(16*now_p)+1]): #그 전 페이지 그림 다시 그리기
            k = canvas.create_line(c - 1, d - 10, c - 1, d - 90, fill="black", width=3) #####
            n = canvas.create_oval(a, b, c, d, fill="black")
            if (b == 160) and (d == 185):
                l = canvas.create_line(a - 10, 170, c + 10, 170, fill="black")
                notes_list[16*(now_p-1)+i] = [n,l,k]
            else:
                notes_list[16*(now_p-1)+i] = [n,k]
            canvas.delete(p1)
            p1 = canvas.create_text(630, 200, text=f"page {now_p}", font=("나눔고딕코딩", 20), fill="blue")
        else:
            #마지막장에 음표가 1개 이상 있을때
            note_pos -= 1

a = notes_list[-1]
if isinstance(a, list):
    for i in a:
        canvas.delete(i) # note
canvas.delete(notes_list[-1])
del notes_list[-1]
del npow_list[-1]
del sound_list[-1]
```

- ▶ remove_one함수는 마지막에 그려진 음표 하나만 지움
- ▶ note_pos가 1이고 마지막 악보 화면에 음표가 1개 있을 때 현재 페이지가 1 이하일 경우 첫번째 페이지라고 알림
- ▶ 이때 마지막 악보 페이지가 1이 아닐 때 그 전 장으로 넘어감
- ▶ 전 장으로 넘어가며 현재 페이지와 전체 페이지 수를 1개 빼고 그 전장의 음표 그림을 다시 그림
- ▶ note_pos가 1이 아닐 경우 음표의 그리는 위치인 note_pos만 1 뺌
- ▶ 삭제된 음표는 객체, 위치좌표, 소리순서 리스트에서 삭제됨



코드설명 main.py



#악보에 음표와 쉼표 그리기

```
def n_draw_save(a,b,c,d):  
    global npos_list, note_pos  
    if (b == 160) and (d == 185): #가온 도 일 경우  
        k = canvas.create_line(c - 1, d - 10, c - 1, d - 90, fill="black", width=3) #####  
        line = canvas.create_line(a - 10, 170, c + 10, 170, fill="black")  
        note = canvas.create_oval(a, b, c, d, fill="black")  
        notes_list.append([note, line, k])  
        npos_list.append([a, b, c, d])  
    else: #가온 도 제외하고 모든 것  
        k = canvas.create_line(c - 1, d - 10, c - 1, d - 90, fill="black", width=3) #####  
        note = canvas.create_oval(a, b, c, d, fill="black")  
        notes_list.append([note, k])  
        npos_list.append([a, b, c, d])  
    note_pos += 1
```

#특정 조건이 되었을 경우 다음 장으로 페이지 넘기기

```
def next_page():  
    global notes_list, note_pos, page, now_p, p1  
    for a in notes_list:  
        if isinstance(a, list):  
            for i in a:  
                canvas.delete(i) # note  
            canvas.delete(a)  
    note_pos = 1  
    page += 1  
    now_p = page  
    canvas.delete(p1)  
    p1 = canvas.create_text(630, 200, text=f"page {page}", font=("나눔고딕코딩", 20), fill="blue")
```

- ▶ n_draw_save함수는 악보에 음표와 쉼표를 그리는 함수
- ▶ 좌표 b가 160, d가 185일 때는 가온도의 위치이므로 타원과 선을 그리고 음표 위치와 그림 객체를 리스트에 저장, note_pos는 1증가
- ▶ next_page함수는 음표가 악보에 꽉 찼을 경우 자동으로 뒷장으로 페이지를 넘기는 함수
- ▶ callback_mouse함수로 음표나 쉼표를 클릭할 때 음표의 위치가 악보의 끝(note_pos=16)일 때 notepos가 17이 되면 다음장으로 넘어감
- ▶ 다음장으로 넘어가며 페이지 수를 변경함

#음표들을 파일로 저장

```
def save_data():  
    global npos_list, sound_list  
    os.makedirs("piano_data", exist_ok=True)  
    npos_list = np.array(npos_list)  
    sound_list = np.array(sound_list)  
    np.save("./piano_data/pos_list1.npy", npos_list)  
    np.save("./piano_data/sound_list1.npy", sound_list)  
    npos_list = npos_list.tolist()  
    sound_list = sound_list.tolist()  
    showinfo("알림", "데이터를 저장하였습니다")
```

#하나씩 음 실행

```
def play_sound(file_name, flag = 0):  
    global sound_list  
    # wav파일 재생  
    path = os.path.join(SOUND_FOLDER, file_name)  
    if flag == 0:  
        sound_list.append(notes[file_name])  
    else:  
        pass  
    playsound(path)
```

def play_all(): #전체 음 실행

```
    global sound_list  
    reverse_dict = dict(map(reversed, notes.items()))  
    for i in sound_list:  
        print(i)  
        if i == 16: #4분쉼표일때  
            time.sleep(0.7)  
            continue  
        elif i == 17: #2분쉼표일때  
            time.sleep(1.4)  
            continue  
        if isinstance(sound_list[0], int): # 불러온 데이터일때  
            name = reverse_dict[i]  
            t = threading.Thread(target=play_sound, args=(name, 1))  
            t.start()  
            time.sleep(0.7)  
        else: # 직접 친 데이터일때  
            t = threading.Thread(target=play_sound, args=(i, 1))  
            t.start()  
            time.sleep(0.7)  
            t.join()
```

코드설명 main.py



- ▶ save_data함수는 음표 위치 좌표와 음 리스트를 npy파일로 저장
- ▶ 파일들은 해당 폴더에 piano_data 폴더를 만들어 그 안에 저장됨
- ▶ play_sound함수는 wav 악기 파일을 실행해 소리가 실행되도록 함
- ▶ play_all함수는 play_sound함수를 이용하여 악보의 전체 음을 한꺼번에 실행
- ▶ 한 음의 1박자를 0.7초로 지정했으므로 4분쉼표일 때는 1박자인 0.7초를, 2분쉼표일 때는 2박자인 1.4초를 씀
- ▶ 불러온 데이터인 경우 숫자로 저장이 되어 딕셔너리로 키와 값을 바꾸어 함수에 넣고, 직접 친 데이터인 경우 파일의 이름으로 저장이 되므로 그냥 함수에 넣어 thread로 동시에 실행이 되도록 함



코드설명 main.py



#저장된 데이터 실행

```
def load_data():
    global npos_list, note_pos, page, now_p, sound_list, p1
    if note_pos == page == now_p == 1:
        data_path = filedialog.askopenfilename(initialdir='./', title='npos_list파일선택', filetypes=((('numpy files', '*.numpy'),
('all files', '*.*'))))
        npos_list = np.load(data_path).tolist()
        data_path = filedialog.askopenfilename(initialdir='./', title='sound_list파일선택', filetypes=((('numpy files',
 '*.numpy'), ('all files', '*.*'))))
        sound_list = np.load(data_path).tolist()

        note_pos = (len(npos_list) % 16) + 1
        page = math.ceil(len(npos_list)/16)
        now_p = 1
        print("~~~~~",page,now_p, note_pos)

    for i, (a,b,c,d) in enumerate(npos_list[(16*(now_p-1)):(16*now_p)]):
        if b == d:
            if sound_list[16 * (now_p - 1) + i] == 16:
                r = canvas.create_image(a, b, anchor=NW, image=rest_4)
            if sound_list[16 * (now_p - 1) + i] == 17:
                r = canvas.create_image(a, b, anchor=NW, image=rest_2)
            # note_pos += 1
            notes_list.append(r)
            continue
        else:
            k = canvas.create_line(c - 1, d - 10, c - 1, d - 90, fill="black", width=3) #####
            n = canvas.create_oval(a, b, c, d, fill="black")
            if (b == 160) and (d == 185):
                l = canvas.create_line(a - 10, 170, c + 10, 170, fill="black")
                notes_list.append([n, l, k])
            else:
                notes_list.append([n, k])
        canvas.delete(p1)
        p1 = canvas.create_text(630, 200, text=f"page {now_p}", font=("나눔고딕코딩", 20), fill="blue")
        showinfo("알림", "데이터를 불러왔습니다")

    else:
        showerror("경고", "음표가 아무것도 저장되어있지 않아야 합니다")
```

- ▶ load_data함수는 save_data함수로 폴더에 저장된 파일을 불러와서 저장하고 악보에 그림
- ▶ 이 함수를 실행하기 위해서는 악보에 아무것도 없어야 함
- ▶ 먼저 파일 선택 창에서 npos_list파일을 먼저 선택하고 두번째로 sound_list파일을 선택
- ▶ 불러온 데이터의 음표 개수로 다음 음표가 그려질 위치 지정, 페이지는 데이터 개수를 16으로 나누었을때 소수점이 있으면 ceil로 숫자를 올려서 지정, 현재 페이지를 첫번째 페이지로 지정
- ▶ 그 다음 악보의 첫번째 페이지에 음들을 그림

```
def callback_mouse(event):
```

```
    global rest_4, rest_2, canvas, note_pos, npos_list, p1, page, now_p, sound_list, instrument
```

```
    # 피아노 변경
```

```
    if (70 < event.x < 230) and (300 < event.y < 360):
```

```
        instrument = 0
```

```
        showinfo("알림", "피아노로 변경하였습니다")
```

```
    # 실로폰 변경
```

```
    if (70 < event.x < 230) and (380 < event.y < 440):
```

```
        instrument = 1
```

```
        showinfo("알림", "실로폰으로 변경하였습니다")
```

```
# 4분침표
```

```
if (70 < event.x < 150) and (460 < event.y < 520):
```

```
    if note_pos % 17 == 0:
```

```
        next_page()
```

```
    r = canvas.create_image(start_pos + (note_gap*note_pos), 80, anchor=NW, image=rest_4)
```

```
    notes_list.append(r)
```

```
    npos_list.append([start_pos + (note_gap*note_pos), 80, start_pos + (note_gap*note_pos), 80])
```

```
    sound_list.append(16)
```

```
    note_pos += 1
```

```
# 2분침표
```

```
if (155 < event.x < 230) and (460 < event.y < 520):
```

```
    if note_pos % 17 == 0:
```

```
        next_page()
```

```
    r = canvas.create_image(start_pos + (note_gap * note_pos)-10, 88, anchor=NW, image=rest_2)
```

```
    notes_list.append(r)
```

```
    npos_list.append([start_pos + (note_gap * note_pos)-10, 88, start_pos + (note_gap * note_pos)-10, 88])
```

```
    sound_list.append(17)
```

```
    note_pos += 1
```

```
# 모든 음표 지우기
```

```
if (960 < event.x < 1180) and (300 < event.y < 360):
```

```
    remove_all()
```

```
# 음표 하나 지우기
```

```
elif (960 < event.x < 1180) and (380 < event.y < 440):
```

```
    remove_one()
```

```
# 음표 데이터 저장
```

```
elif (960 < event.x < 1180) and (460 < event.y < 540):
```

```
    save_data()
```

```
# 음표 데이터 불러오기
```

```
elif (960 < event.x < 1180) and (540 < event.y < 620):
```

```
    load_data()
```

```
# 음들을 전체 실행하기
```

```
elif (960 < event.x < 1180) and (620 < event.y < 700):
```

```
    t = threading.Thread(target=play_all)
```

```
    t.start()
```

코드 설명

main.py – callback_mouse 함수

- ▶ callback_mouse 함수는 마우스로 canvas에 왼쪽 클릭을 했을 때 실행됨
- ▶ 악기 변경 버튼의 범위를 지정하고 눌렀을 때 악기 flag인 instrument에 다른 숫자를 지정해 주어 악기를 변경
- ▶ 4분침표와 2분침표 버튼의 범위를 지정하고 피아노 음과 같이 악보에 침표 사진을 그리고 리스트들에 값들을 저장함
- ▶ 이 외의 5개의 기능들은 앞서 설명한 함수들을 불러 기능을 실행함
- ▶ 음들을 전체 실행할 때는 소리와 인터페이스가 동시에 진행되어야 하므로 thread사용



코드설명

main.py – callback_mouse 함수

악보 오른쪽으로 넘기기

```
elif (1210 < event.x < 1245) and (85 < event.y < 150):
```

```
    if now_p == page:
```

```
        showwarning("경고", "다음 장이 없습니다")
```

```
        return
```

```
    now_p += 1
```

```
    print("오른쪽#####", now_p, (16*(now_p-1)), 16*now_p)
```

```
    for a in notes_list[16*((now_p-1)-1):(16*(now_p-1)+1)]: #전 장 지우기
```

```
        if isinstance(a, list):
```

```
            for i in a:
```

```
                canvas.delete(i) # note
```

```
    else:
```

```
        canvas.delete(a)
```

```
try: #마지막 페이지가 꼭 차있지 않을 경우에 index error남
```

```
    for i, (a, b, c, d) in enumerate(npos_list[(16 * (now_p - 1)):(16 * now_p)]): # 현재 장 그리기
```

```
        if b == d:
```

```
            if sound_list[16 * (now_p - 1) + i] == 16:
```

```
                r = canvas.create_image(a, b, anchor=NW, image=rest_4)
```

```
            if sound_list[16 * (now_p - 1) + i] == 17:
```

```
                r = canvas.create_image(a, b, anchor=NW, image=rest_2)
```

```
            #note_pos += 1
```

```
            notes_list.append(r)
```

```
            continue
```

```
    else:
```

```
        print("iiii", npos_list[16 + i], npos_list[(16 * (now_p - 1)):(16 * now_p)])
```

```
        k = canvas.create_line(c - 1, d - 10, c - 1, d - 90, fill="black", width=3) #####
```

```
        n = canvas.create_oval(a, b, c, d, fill="black")
```

```
        if (b == 160) and (d == 185):
```

```
            l = canvas.create_line(a - 10, 170, c + 10, 170, fill="black")
```

```
            notes_list[16 * (now_p - 1) + i] = [n, l, k]
```

```
    else:
```

```
        notes_list[16 * (now_p - 1) + i] = [n, k]
```

```
except IndexError:
```

```
    canvas.delete(p1)
```

```
    p1 = canvas.create_text(630, 200, text=f"page {now_p}", font=("나눔고딕코딩", 20), fill="blue")
```

```
    return
```

```
canvas.delete(p1)
```

```
p1 = canvas.create_text(630, 200, text=f"page {now_p}", font=("나눔고딕코딩", 20), fill="blue")
```

- ▶ 악보를 오른쪽으로 넘기는 버튼의 범위를 지정하고 현재 페이지와 전체 페이지가 같을 경우 경고문을 알림
- ▶ 다음 페이지가 존재할 경우 지금 장을 먼저 지운 다음 다음 페이지를 그림
- ▶ 리스트로 악보 페이지를 불러오는데 마지막 페이지가 꼭 차있지 않으면 리스트에 값이 없으므로 에러가 발생함. 그러므로 try except문으로 예외를 지정
- ▶ 심표를 그릴 경우 사진을 그리는 것이므로 좌표 2개만 필요. 그래서 같은 값을 두번 저장했으므로 b와 d값이 같을 경우 심표를 그린 것으로 판단
- ▶ 이때 notes 딕셔너리에서 4분심표는 16, 2분심표는 17로 지정했으므로 이것으로 심표 종류 파악
- ▶ 이 외의 경우 음표를 그린 것으로 판단하고 페이지 글씨를 변경



코드 설명

main.py – callback_mouse 함수

악보 왼쪽으로 넘기기

```
elif (35 < event.x < 70) and (85 < event.y < 120):
```

```
    if now_p <= 1:
```

```
        showwarning("경고", "첫 번째 페이지입니다")
```

```
        return
```

```
    now_p -= 1
```

```
for a in notes_list[16*now_p:16*(now_p+1)]: #전 장 지우기
```

```
    if isinstance(a, list):
```

```
        for i in a:
```

```
            canvas.delete(i) # note
```

```
    else:
```

```
        canvas.delete(a)
```

```
for i,(a,b,c,d) in enumerate(npos_list[16*(now_p-1):(16*now_p)]): #현재 장 그리기
```

```
    if b == d:
```

```
        if sound_list[16 * (now_p - 1) + i] == 16:
```

```
            r = canvas.create_image(a, b, anchor=NW, image=rest_4)
```

```
        if sound_list[16 * (now_p - 1) + i] == 17:
```

```
            r = canvas.create_image(a, b, anchor=NW, image=rest_2)
```

```
        notes_list[16 * (now_p - 1) + i] = r
```

```
        continue
```

```
    else:
```

```
        k = canvas.create_line(c - 1, d - 10, c - 1, d - 90, fill="black", width=3) #####
```

```
        n = canvas.create_oval(a, b, c, d, fill="black")
```

```
        if (b == 160) and (d == 185):
```

```
            l = canvas.create_line(a - 10, 170, c + 10, 170, fill="black")
```

```
            notes_list[16 * (now_p - 1) + i] = [n, l, k]
```

```
        else:
```

```
            notes_list[16 * (now_p - 1) + i] = [n, k]
```

```
print(notes_list)
```

```
canvas.delete(p1)
```

```
p1 = canvas.create_text(630, 200, text=f"page {now_p}", font=("나눔고딕코딩", 20), fill="blue")
```

- ▶ 악보를 왼쪽으로 넘기는 버튼의 범위를 지정하고 현재 페이지가 1이거나 작을 때 경고문을 알림
- ▶ 현재 페이지가 1이 아닐 경우 먼저 그 전 페이지로 변수를 감소시키고 전 장의 그림들을 지움
- ▶ 그 다음 현재 장을 그림. b와 d가 같은 경우에는 쉼표를 그릴 때 밖에 없으므로 i로 지정하였고 아닐 경우에는 음표를 그리고 리스트에 객체를 저장함
- ▶ 이때 notes 딕셔너리에서 4분쉼표는 16, 2분쉼표는 17로 지정했으므로 이것으로 쉼표 종류 파악



코드 설명

main.py – callback_mouse 함수

```
# 건반
elif 300 < event.y < 600:
    if 270 < event.x < 350: #도
        if note_pos % 17 == 0:
            next_page()

        n_draw_save(start_pos + (note_gap*note_pos), 160, last_pos + (note_gap*note_pos), 185)
        if instrument == 0:
            note = "do-c4.wav"
        else:
            note = "xylophone-c1.wav"
        t = threading.Thread(target=play_sound, args=([note])) #동시에 음계 소리 실행
        t.start()

elif 350 < event.x < 430: #리/
    if note_pos % 17 == 0:
        next_page()

    n_draw_save(start_pos + (note_gap*note_pos), 148, last_pos + (note_gap*note_pos), 173) #음표 그리기
    if instrument == 0:
        note = "re-d4.wav"
    else:
        note = "xylophone-d1.wav"
    t = threading.Thread(target=play_sound, args=([note])) #동시에 음계 소리 실행
    t.start()

elif 430 < event.x < 510: #미/
    if note_pos % 17 == 0:
        next_page()

    n_draw_save(start_pos + (note_gap*note_pos), 136, last_pos + (note_gap*note_pos), 161) #음표 그리기
    if instrument == 0:
        note = "mi-e4.wav"
    else:
        note = "xylophone-e1.wav"
    t = threading.Thread(target=play_sound, args=([note])) #동시에 음계 소리 실행
    t.start()
```

- ▶ 피아노 건반들은 범위를 지정하여 위치를 지정함
- ▶ 건반들의 구성은 모두 같음
- ▶ 먼저 만약 note_pos가 17로 다음장으로 넘어가야 하면 next_page함수를 불러 다음 장으로 간 후에 음표를 그리게 됨
- ▶ 아닐 경우에는 악보에 음표를 그린 후 피아노 일때와 실로폰 일 때의 경우의 수에 따라 다른 값을 play_sound에 넣음
- ▶ 악기의 소리가 실행됨과 동시에 악보에 그려져야 하기 때문에 thread사용



코드 설명

main.py – callback_mouse 함수

elif 510 < event.x < **590**: #파

```
if note_pos % 17 == 0:
    next_page()
```

n_draw_save(start_pos + (note_gap*note_pos), **124**, last_pos + (note_gap*note_pos), **149**) #음표 그리기

#note_pos += 1

```
if instrument == 0:
    note = "fa-f4.wav"
```

else:

note = "xylophone-f1.wav"

t = threading.Thread(target=play_sound, args=([note])) #동시에 음계 소리 실행

t.start()

elif 590 < event.x < **670**: #솔

```
if note_pos % 17 == 0:
    next_page()
```

n_draw_save(start_pos + (note_gap*note_pos), **112**, last_pos + (note_gap*note_pos), **137**) #음표 그리기

#note_pos += 1

```
if instrument == 0:
    note = "sol-g4.wav"
```

else:

note = "xylophone-g1.wav"

t = threading.Thread(target=play_sound, args=([note])) #동시에 음계 소리 실행

t.start()

elif 670 < event.x < **750**: #라

```
if note_pos % 17 == 0:
    next_page()
```

n_draw_save(start_pos + (note_gap*note_pos), **100**, last_pos + (note_gap*note_pos), **125**) #음표 그리기

#note_pos += 1

```
if instrument == 0:
    note = "la-a4.wav"
```

else:

note = "xylophone-a1.wav"

t = threading.Thread(target=play_sound, args=([note])) #동시에 음계 소리 실행

t.start()

elif 750 < event.x < **830**: #시

```
if note_pos % 17 == 0:
    next_page()
```

n_draw_save(start_pos + (note_gap*note_pos), **88**, last_pos + (note_gap*note_pos), **113**) #음표 그리기

#note_pos += 1

```
if instrument == 0:
    note = "si-b4.wav"
```

else:

note = "xylophone-b1.wav"

t = threading.Thread(target=play_sound, args=([note])) #동시에 음계 소리 실행

t.start()

elif 830 < event.x < **910**: #도

```
if note_pos % 17 == 0:
    next_page()
```

n_draw_save(start_pos + (note_gap*note_pos), **76**, last_pos + (note_gap*note_pos), **101**) #음표 그리기

#note_pos += 1

```
if instrument == 0:
    note = "do-c5.wav"
```

else:

note = "xylophone-c2.wav"

t = threading.Thread(target=play_sound, args=([note])) #동시에 음계 소리 실행

t.start()

▶ 앞의 형식과 같으므로 생략

앞으로의 개선방안

- ▶ 1. 모든 음표들과 쉼표들을 추가할 예정
- ▶ 2. 음 파일을 더 구해서 가온도에서 높은 도까지가 아닌 더 넓은 폭의 음을 낼 수 있도록 할 예정
- ▶ 3. 다양한 악기의 파일들을 구해 악기들을 추가할 예정



감사합니다.

