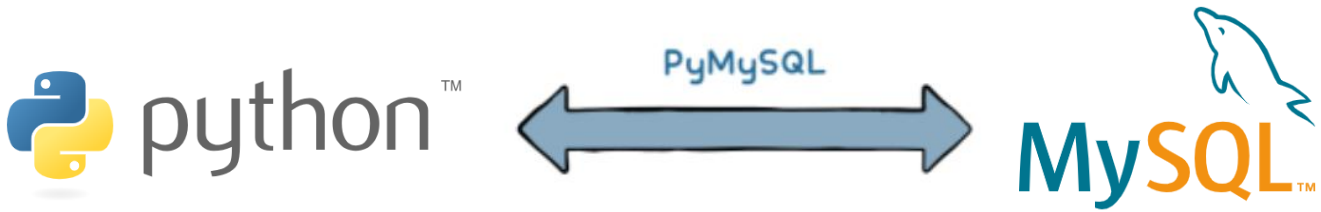
Database

8장 SQL과 파이썬의 연결

동아대학교 컴퓨터공학과
양 선

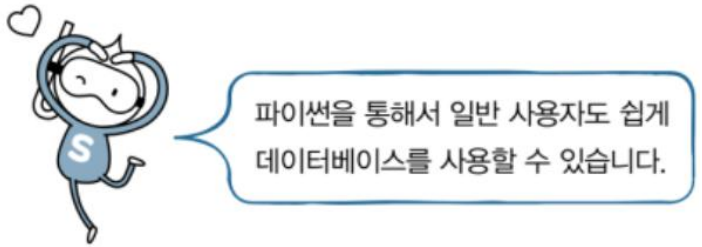


파이썬 개발 환경 준비 (교재 p.372)

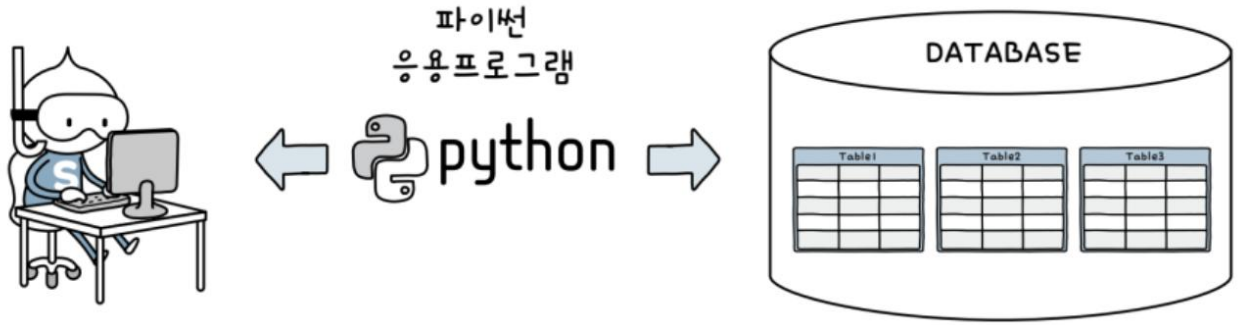
◦ 파이썬 장점
- 무료로 강력한 기능 사용
- 설치와 사용 환경 구축이 쉽고
- 다양하고 강력한 외부 라이브러리들이 많다

◦ 설치
- IDLE 스크립트 모드 사용해도 되고, 다른 개발툴 사용하셔도 됩니다.
- 우리는 PyMySQL 사용해서 파이썬과 MySQL 연결
  - 예를 들어 IDLE 사용시 pip install pymysql
  - 아나콘다 사용시 conda install pymysql
  - 기타 본인의 개발툴에 맞춰서 pymysql을 인스톨하세요.

파이썬과 MySQL 연동



- 연동이 완성되면 커맨드 라인이나 워크벤치 없어도 MySQL에 접근 가능합니다.



- 새로운 데이터베이스 만들어도 되지만 우리는 기존에 사용하던 shop_db를 이용하겠습니다.
- 이제 커맨드 라인이나 워크벤치는 파이썬 연동이 잘 되고있는지에 대한 확인용 정도로만 사용하겠습니다.

테이블 생성 예제

```
import pymysql

connect = pymysql.connect(host='127.0.0.1', user='root',
                          password='1234', db='shop_db', charset='utf8')

cursor = connect.cursor()

sql = "create table usertable  \
      (userid char(10), username char(10), email char(20), regyear int)"

cursor.execute(sql)

connect.commit()         # DDL은 대부분 암시적 커밋

connect.close()

print('테이블 생성 완료!' )
```

insert 예제

```
import pymysql
connect = pymysql.connect(host='127.0.0.1', user='root',
                          password='1234', db='shop_db', charset='utf8')
cursor = connect.cursor()
count = 0
while (True) :
    data1 = input("사용자 ID (Q:종료)==> ")
    if data1 == "q" or data1 == "Q" :
        break;
    data2 = input("사용자 이름 ==> ")
    data3 = input("사용자 이메일 ==> ")
    data4 = input("가입 연도 (정수) ==> ")
    sql = "INSERT INTO  usertable VALUES \
           ('" + data1 + "','" + data2 + "','"  + data3 + "',"+ data4 + ")"
    cursor.execute(sql)
    count += 1
    print('--------------')

connect.commit()                # 데이터가 바뀌는 DML은 커밋 해줘야 실제 저장됩니다.
connect.close()
print(count, ' 건 등록 완료')
```

select 예제

```
import pymysql
connect = pymysql.connect(host='127.0.0.1', user='root',
                          password='1234', db='shop_db', charset='utf8')
cursor = connect.cursor()

cursor.execute("SELECT * FROM usertable")

print("사용자ID     사용자이름      이메일                  가입연도")
print("-------------------------------------------------------")

while (True) :
    row = cursor.fetchone()                 # select 해 온 각 행(튜플)
    if row== None :
        break
    print("%-10s   %-10s   %-20s   %d" % (row[0], row[1], row[2], row[3]))

connect.close()
```

[중간 정리]

▶ 6가지 키워드로 끝내는 핵심 포인트

- **데이터베이스 연동**은 SQL을 파이썬과 연결하는 것을 말합니다.
- **import pymysql** 명령을 사용해서 SQL과 파이썬을 연동합니다.
- MySQL과 파이썬의 연결을 위해서는 pymysql.connect()로 **연결자**를 생성하고, 연결 통로인 **커서**를 통해 파이썬에서 MySQL로 SQL을 전송합니다.
- 데이터를 변경(예: 입력)한 후에는 **커밋**을 수행해야 변경된 내용이 확정됩니다.
- 파이썬에서 SELECT 문으로 데이터를 조회한 후에는 **fetchone()** 함수를 통해서 데이터를 한 행씩 가져옵니다.

[확인문제]

1. 다음은 파이썬에서 MySQL로 데이터를 입력하기 위한 설명입니다. 가장 거리가 먼 것을 하나 고르세요.

① 연결자는 connect() 함수를 사용합니다.
② 테이블 생성은 execute("CREATE TABLE~~")을 사용합니다.
③ 데이터 입력은 execute("INSERT 문장")을 사용합니다.
④ close()로 연결자를 종료한 후에는 commit()으로 저장해야 합니다.

2. 다음은 pymysql.connect() 함수의 옵션에 대한 설명입니다. 가장 거리가 먼 것을 하나 고르세요.

① host에는 IP 주소를 적어줍니다.
② user에는 사용자를 적어줍니다.
③ password에는 암호를 적어줍니다.
④ charset에는 데이터베이스를 적어줍니다.

[확인문제]

3. 다음은 파이썬과 MySQL 연동을 위한 코드입니다. 가장 거리가 먼 것을 하나 고르세요.

① 연결자.cursor()는 커서를 생성합니다.

② 커서.execute("CREATE TABLE~~")은 테이블을 생성합니다.

③ 커서.insert("INSERT INTO~~")는 데이터를 입력합니다.

④ 연결자.commit()은 입력한 내용을 저장합니다.

4. 조회한 결과를 한 행씩 접근하는 파이썬 함수를 고르세요.

① selectone()

② fetchone()

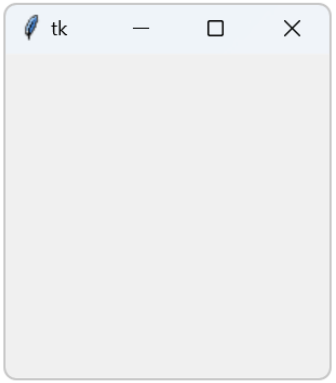
③ oneline()

④ ditpatch()

파이썬 GUI 기본 화면 (교재 p. 396)

◦ 첫 화면 띄우기

```
import tkinter

window = tkinter.Tk()

# 여기 코딩

window.mainloop()
```



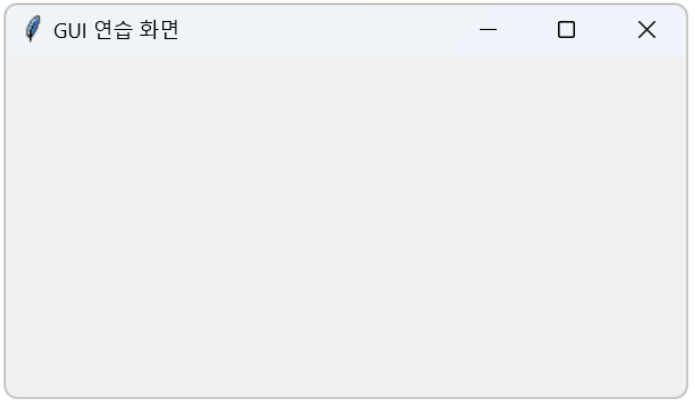
❶ tkinter는 파이썬에서 GUI 관련 모듈을 제공해주는 표준 윈도 라이브러리입니다. 윈도 화면이 필요할 때는 꼭 써줘야 합니다. (install 이 필요한 외부 라이브러리 아님)

❷ Tk()는 기본이 되는 윈도를 반환하는데, 이를 wondow 변수에 넣었습니다. Tk()를 루트 윈도 root window라고도 부르며, 꼭 필요한 요소입니다. 이 행이 실행될 때 윈도가 출력됩니다.

❸ wondow.mainloop() 함수를 실행합니다. 이는 앞으로 윈도에 키보드 누르기, 마우스 클릭 등의 다양한 작업이 일어날 때 이벤트를 처리하기 위해 필요한 부분입니다.

윈도 제목과 크기

```
import tkinter
window = tkinter.Tk()

window.title('GUI 연습 화면')
window.geometry('400x200')

window.mainloop()
```



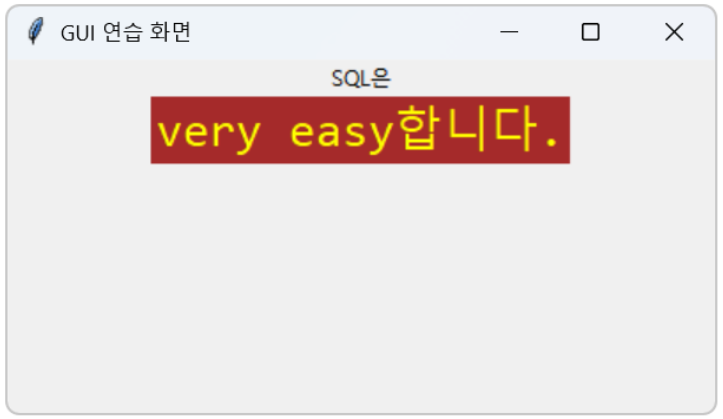
- title()은 윈도에 타이틀 부여
- geometry()는 초기 윈도 사이즈인데, 따옴표 주의! 중간에 * 아니고 x 인 거 주의!

위젯

- 이제 몇몇 위젯widget을 화면에 표시해 보겠습니다.
  - 위젯은 윈도에 나오는 텍스트, 버튼, 라디오, 이미지 등을 통합해서 지칭하는 용어
  - 모든 위젯들은 pack() 함수를 사용해야 화면에 나타납니다.

- 먼저 라벨label부터 표시해 보겠습니다.
  - 라벨은 문자를 표현할 수 있는 위젯으로 label(부모윈도, 옵션...) 형식을 사용합니다.

Label() 함수




```
import tkinter
window = tkinter.Tk()
window.title('GUI 연습 화면')
window.geometry('400x200')

a = tkinter.Label(window, text='SQL은')
b = tkinter.Label(window, text='very easy합니다.',
                          font=('consolas', 20), bg='brown', fg='yellow')
a.pack()
b.pack()

window.mainloop()
```

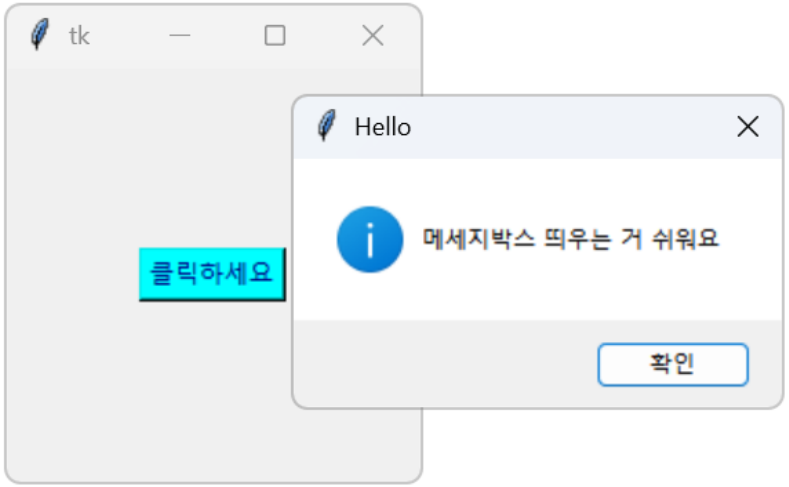
❶ Label() 함수는 라벨을 만듭니다. 옵션에서 text는 글자 내용을, font는 글꼴과 크기를 지정합니다. fg는 foreground의 약자로 글자색을, bg는 background의 약자로 배경색을 지정합니다.

❷ pack() 함수를 통해서 해당 라벨을 화면에 표시해준 것입니다. label(부모 윈도, 옵션 ...)에서 부모 윈도는 루트 윈도인 root를 지정했습니다. 그러므로 루트 윈도에 라벨이 출력됩니다.

라벨(label)은 글자를 다양한 형태로 표현합니다.

Button() 함수

```
import tkinter
import tkinter.messagebox as mbox

def function1() :
    mbox.showinfo('Hello', '메세지박스 띄우는 거 쉬워요')

window = tkinter.Tk()
window.geometry('200x200')

a = tkinter.Button(window, text='클릭하세요',
                   fg='navy', bg='cyan', command=function1)
a.pack(expand = 1)

window.mainloop()
```



❶ 메시지 상자를 사용하기 위해서 import했습니다.

❷ command 옵션에 function1() 함수를 호출했습니다. 버튼을 클릭하면 메시지 상자가 나타납니다.

버튼(Button)은 클릭했을 때 실행하는 기능을 구현합니다.

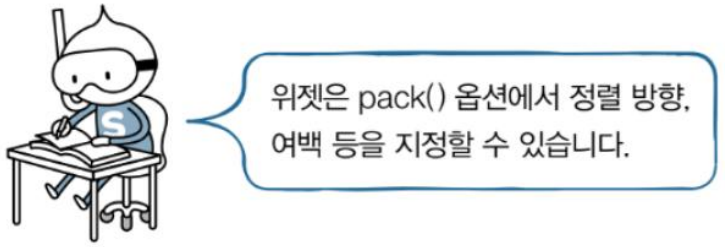
❸ pack()에서 버튼을 화면 중앙에 표현하기 위해 expand=1 옵션을 추가했습니다.

위젯 정렬

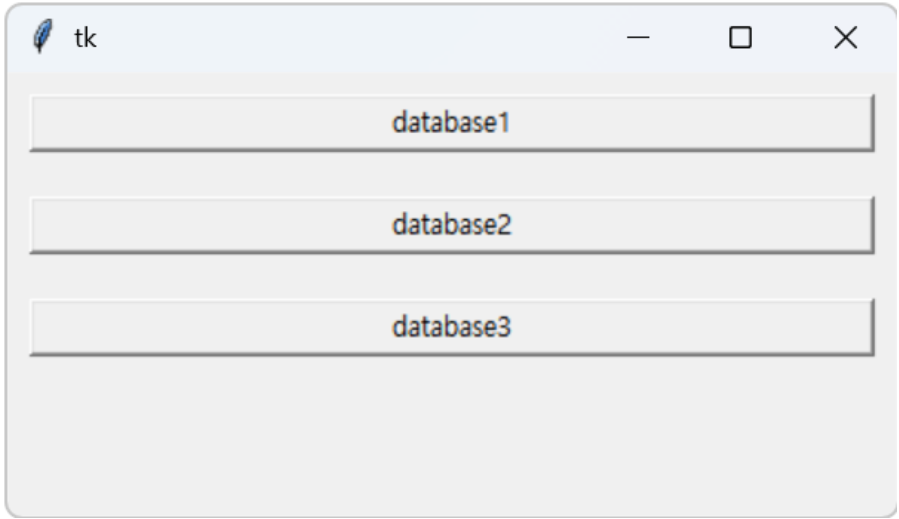
```
import tkinter
window = tkinter.Tk()
window.geometry('400x200')

a = tkinter.Button(window, text='database1')
b = tkinter.Button(window, text='database2')
c = tkinter.Button(window, text='database3')

a.pack(side='left')
b.pack(side='left')
c.pack(side='left')

window.mainloop()
```

- side='left' 로 하면 왼쪽부터 채워나갑니다.
- right, top, bottom도 해보세요.

pack() 속성: fill, padx, pady




```
import tkinter
window = tkinter.Tk()
window.geometry('400x200')

a = tkinter.Button(window, text='database1')
b = tkinter.Button(window, text='database2')
c = tkinter.Button(window, text='database3')

a.pack(side='top', fill='x', padx=10, pady=10)
b.pack(side='top', fill='x', padx=10, pady=10)
c.pack(side='top', fill='x', padx=10, pady=10)

window.mainloop()
```



- side가 top일 때 fill='x' 수평으로 늘리기 (x, y, both)
  - side를 left로 하고 fill='y' 해보세요.
- padx, pady는 수평 수직으로 여백 부여

프레임, 엔트리, 리스트박스

```
import tkinter
window = tkinter.Tk()
window.geometry('200x250')

upFrame = tkinter.Frame(window)
downFrame = tkinter.Frame(window)
upFrame.pack()
downFrame.pack()

box1 = tkinter.Entry(upFrame, width = 10)
box1.pack(padx = 20, pady = 20)
box2 = tkinter.Listbox(downFrame, bg = 'yellow')
box2.pack()

box2.insert('end', '하나')
box2.insert('end', '둘')
box2.insert('end', '셋')

window.mainloop()
```

❶ upFrame 및 downFrame이라는 2개의 프레임을 생성하고 화면에 출력했습니다. 프레임은 구역을 나눈 것일 뿐 화면에는 표시되지 않습니다.

❷ 입력을 위한 엔트리를 나타냈습니다. 그런데 이번에는 root가 아닌 upFrame에 나오도록 했습니다.

❸ 리스트 박스를 나타냈습니다. 리스트 박스는 아래쪽인 downFrame에 나오도록 했습니다.

❹ 리스트 박스에 데이터를 3건 입력했습니다. 옵션 중 END는 데이터를 제일 뒤에 첨부하라는 의미입니다. 그래서 차례대로 하나, 둘, 셋이 리스트 박스에 나왔습니다.

usertable 화면 예제 (1/3)

```
import pymysql
import tkinter
import tkinter.messagebox as mbox

def insertData() :

    connect = pymysql.connect(host='127.0.0.1', user='root',
                      password='1234', db='shop_db', charset='utf8')
    cursor = connect.cursor()

    data1 = edt1.get();    data2 = edt2.get();    data3 = edt3.get();    data4 = edt4.get()
    sql = "INSERT INTO  usertable VALUES(%s, %s, %s, %s)"
    data = (data1, data2, data3, data4)
    cursor.execute(sql, data)

    connect.commit()
    connect.close()

    mbox.showinfo('성공', '1건 등록 완료')
    selectData()
```

usertable 화면 예제 (2/3)

```
def selectData() :
    strData1, strData2, strData3, strData4  = [], [], [], []
    strData1.append("사용자 ID");      strData2.append("사용자 이름")
    strData3.append("이메일");         strData4.append("출생연도 (숫자)")
    strData1.append("------------");   strData2.append("------------")
    strData3.append("------------");   strData4.append("------------")

    connect = pymysql.connect(host='127.0.0.1', user='root', password='1234', db='shop_db', charset='utf8')
    cursor = connect.cursor()
    cursor.execute("SELECT * FROM userTable order by userid")

    while (True) :
        row = cursor.fetchone()
        if row== None :  break;
        strData1.append(row[0]);  strData2.append(row[1]);  strData3.append(row[2]);  strData4.append(row[3])

    listData1.delete(0,listData1.size() - 1);   listData2.delete(0,listData2.size() - 1)
    listData3.delete(0,listData3.size() - 1);   listData4.delete(0,listData4.size() - 1)

    for item1, item2, item3, item4 in zip(strData1, strData2, strData3, strData4 ):
        listData1.insert('end', item1);      listData2.insert('end', item2)
        listData3.insert('end', item3);      listData4.insert('end', item4)
    connect.close()
```

usertable 화면 예제(3/3)

```
## 메인 코드부
window = tkinter.Tk()
window.geometry("600x300")
window.title('완전한 GUI 응용 프로그램')

edtFrame = tkinter.Frame(window);  edtFrame.pack()
listFrame = tkinter.Frame(window);  listFrame.pack(side ='bottom',fill='both', expand=1)

edt1= tkinter.Entry(edtFrame, width=10);    edt1.pack(side='left',padx=10,pady=10)
edt2= tkinter.Entry(edtFrame, width=10);    edt2.pack(side='left',padx=10,pady=10)
edt3= tkinter.Entry(edtFrame, width=10);    edt3.pack(side='left',padx=10,pady=10)
edt4= tkinter.Entry(edtFrame, width=10);    edt4.pack(side='left',padx=10,pady=10)

btnInsert = tkinter.Button(edtFrame, text="입력", command = insertData)
btnInsert.pack(side='left',padx=10,pady=10)
btnSelect = tkinter.Button(edtFrame, text="조회", command =selectData )
btnSelect.pack(side='left',padx=10,pady=10)

listData1 = tkinter.Listbox(listFrame,bg = 'yellow');   listData1.pack(side='left',fill='both', expand=1)
listData2 = tkinter.Listbox(listFrame,bg = 'yellow');   listData2.pack(side='left',fill='both', expand=1)
listData3 = tkinter.Listbox(listFrame,bg = 'yellow');   listData3.pack(side='left',fill='both', expand=1)
listData4 = tkinter.Listbox(listFrame,bg = 'yellow');   listData4.pack(side='left',fill='both', expand=1)
window.mainloop()
```

[중간 정리]

▶ 7가지 키워드로 끝내는 핵심 포인트

- **GUI**는 윈도를 제공함으로써 사용자가 편리하게 데이터베이스에 접근하도록 도와주는 환경을 말합니다.
- **tkinter**는 파이썬에서 GUI를 만들기 위해 제공되는 라이브러리입니다.
- **라벨**은 윈도에 문자를 표현하고, **버튼**은 클릭하는 기능을 제공합니다.
- **프레임**은 화면을 나누는 기능이고, **엔트리**는 입력 상자를 제공합니다. **리스트 박스**는 여러 건의 목록을 표현하는 기능입니다.

[확인문제]

1. 다음은 파이썬에 기본 윈도를 생성하기 위한 설명입니다. 가장 거리가 먼 것을 하나 고르세요.

① tkinter를 임포트해야 합니다.
② 루트 윈도는 Tk() 함수를 사용해서 생성합니다.
③ title() 함수로 윈도의 크기를 조절합니다.
④ mainloop() 함수를 사용해서 사용자 이벤트를 처리합니다.

2. 다음은 라벨과 관련된 내용입니다. 가장 거리가 먼 것을 하나 고르세요.

① Label() 함수로 라벨을 생성합니다.
② text 옵션은 라벨의 글자 내용입니다.
③ font는 글꼴이나 크기를 지정합니다.
④ bg는 글자의 색상을 지정합니다.

[확인문제]

3. 다음은 버튼과 관련된 내용입니다. 가장 거리가 먼 것을 하나 고르세요.

① Button() 함수로 버튼을 생성합니다.

② text 옵션은 버튼의 글자 내용입니다.

③ function은 실행 함수명을 지정합니다.

④ fg는 글자의 색상을 지정합니다.

4. 다음은 위젯을 정렬하는 side 옵션의 값입니다. 관련이 없는 것을 고르세요.

① LEFT

② RIGHT

③ UP

④ BOTTOM