

코딩, Python으로 같이 시작해요



과제 설명 1

- 아래의 Person 클래스에 age 속성을 추가하세요.(속성은 생성자 안에서 만들어야 하겠지요?)
- Person 클래스에 자기소개를 하는 greeting 메서드를 추가하세요.
(자기 이름과 나이를 말해야 합니다.)

```
class Person:
    def __init__(self, name, energy, age):
        self.name = name
        self.energy = energy
        self.age = age

    def eat(self):
        print("밥을 먹어 기운이 납니다.")
        self.energy += 10
        print("{0}님의 현재 에너지 : {1}".format(self.name, self.energy))

    def greeting(self):
        print("안녕하세요, 저는 {0}입니다. {1}살 입니다.".format(self.name, self.age))
```

과제 설명 2

- (실습에서 만들었던)Account 클래스에 돈을 출금하는 withdraw 메서드를 추가하세요.
- 출금하려는 액수가 남은 잔액(money 속성 -> self.money)보다 크면 “잔액이 부족합니다.”라고 문구를 띄우고 잔액을 알려주세요.
- 잔액이 충분하면 self.money에서 해당 액수를 차감하고 남은 잔액을 알려주세요.

```
class Account:
    def __init__(self, name, money):
        self.name = name
        self.money = money

    def deposit(self, amount):
        self.money += amount
        print("-----{0}님의 계좌-----".format(self.name))
        print("{0}원을 입금했습니다.".format(amount))
        print("잔액 : {0}".format(self.money))

    def withdraw(self, amount):
        print("-----{0}님의 계좌-----".format(self.name))
        if amount <= self.money:
            self.money -= amount
            print("{0}를 출금했습니다.\n잔액 : {1}".format(amount, self.money))
        else:
            print("잔액이 부족합니다.\n잔액 : {0}".format(self.money))
```

클래스 상속

<클래스 상속>

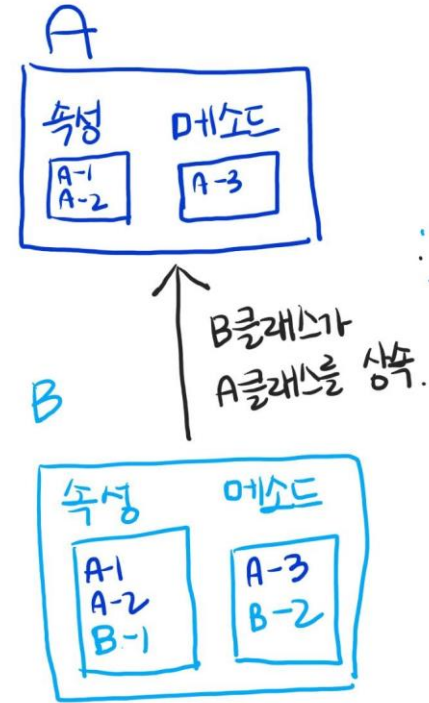
- : 클래스 상속은 다른 클래스의 요소들을 똑같이 사용하면서 새로운 기능을 추가하고 싶을 때 사용합니다.
- : B클래스가 A 클래스를 상속 받으면, B 클래스는 A 클래스에 있는 모든 메소드와 속성들을 똑같이 사용할 수 있고, 자신에게 필요한 메소드나 속성을 추가할 수 있는 것이죠.
- : 물려주는 클래스를 **부모 클래스(기반 클래스)**, 물려받는 클래스를 **자식 클래스(파생 클래스)**라고 부릅니다.

<클래스 상속 방법>

class 클래스_이름(상속하는 클래스 이름):

<클래스는 언제 사용?>

- : 클래스는 **is a 관계**를 만족할 때 사용하는 것이 적합합니다.
- ‘자식 클래스 is a 부모 클래스’ 라고 말할 수 있는 관계



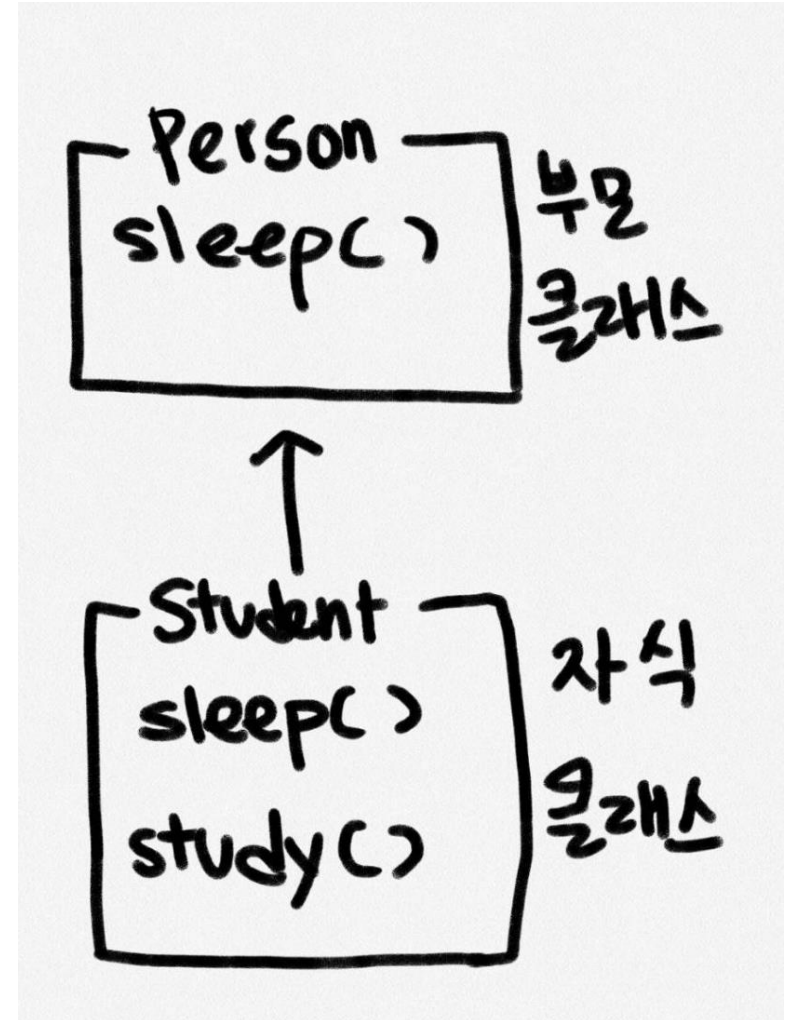
클래스 상속

Person 클래스를 상속받은 Student 클래스는 자신의 메소드 study 뿐만 아니라, 부모 클래스의 메소드 sleep도 사용할 수 있습니다.

```
class Person:
    def sleep(self):
        print("저는 사람이라서 잠을 잡니다.")

class Student(Person):
    def study(self):
        print("저는 학생이라서 공부도 합니다.")

>>> james = Student()
>>> james.sleep()
저는 사람이라서 잠을 잡니다.
>>> james.study()
저는 학생이라서 공부도 합니다.
```



클래스 상속

앞 예제에서는 부모 클래스의 메소드만 가져다 썼는데요,
이번에는 자식 클래스에서 기반 클래스의 속성들도 가져다 써보겠습니다.

```
class Person:
    def __init__(self):
        print("Person 클래스의 생성자가 호출 되었습니다.")
        self.name = "길동이"

class Student(Person):
    def __init__(self):
        print("Student 클래스의 생성자가 호출 되었습니다.")
        self.school = "호그와트 마법 학교"
```

```
>>> student1 = Student()
Student 클래스의 생성자가 호출 되었습니다.
>>> student1.school
'호그와트 마법 학교'
>>> student1.name
Traceback (most recent call last):
  File "<pyshell#3>", line 1, in <module>
    student1.name
AttributeError: 'Student' object has no attribute 'name'
>>> student1.__dict__
{'school': '호그와트 마법 학교'}
```

Student 클래스의 인스턴스 student1에서 부모 클래스의 속성인 name은 만들어지지 않았습니다. student1 객체를 만들 때 Person 클래스의 생성자가 호출되지 않았기 때문입니다. 다음 슬라이드에서 이 문제를 해결해 봅시다.

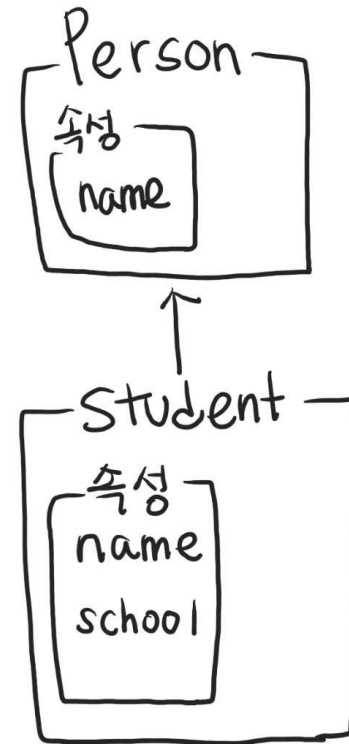
클래스 상속

- 자식 클래스를 만들 때는 부모 클래스의 생성자를 호출하는 코드를 따로 명시해줘야 합니다.
- **super().부모 클래스의 메소드()**
 - : super()를 사용해 생성자를 포함한 부모 클래스의 메소드를 불러올 수 있습니다.
 - : super().__init__() 로 부모 클래스의 생성자를 호출하면 부모 클래스의 속성 까지 모두 정상적으로 만들어집니다.

```
class Person:
    def __init__(self):
        print("Person 클래스의 생성자가 호출 되었습니다.")
        self.name = "길동이"

class Student(Person):
    def __init__(self):
        print("Student 클래스의 생성자가 호출 되었습니다.")
        super().__init__()
        self.school = "호그와트 마법 학교"
```

```
>>> student1 = Student()
Student 클래스의 생성자가 호출 되었습니다.
Person 클래스의 생성자가 호출 되었습니다.
>>> student1.__dict__
{'name': '길동이', 'school': '호그와트 마법 학교'}
>>> student1.name
'길동이'
```



PyQt란

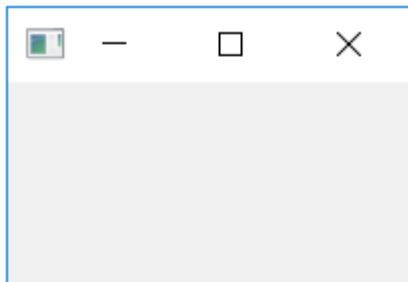
- GUI
 - : 그림이나 색상 같은 그래픽 요소를 조작해서 컴퓨터를 제어하는 인터페이스
 - : 예시) 윈도우 바탕화면
- QT
 - : C++ 언어 기반 GUI 프레임워크
 - : GUI 프로그램을 만들기 위한 뼈대!
- PyQt
 - : QT를 파이썬에서 사용할 수 있도록 만든 것.
 - : 정리하자면, PyQt는 그림을 눌러서 컴퓨터를 제어하는 GUI 프로그램을 만들기 위한 파이썬 모듈.
- 모듈?
 - : 모듈은 함수, 변수 또는 클래스를 모아놓은 파일로, 다른 사람들이 만들어 놓은 모듈을 불러와 사용할 수 있다.
 - : PyQt 설치를 완료했기 때문에 import를 사용해 PyQt5 모듈을 불러와 그 안에 있는 클래스들을 사용할 수 있습니다.

기본 창 띄우기

```
import sys
from PyQt5.QtWidgets import *

class WindowClass(QMainWindow):
    def __init__(self):
        super().__init__()

if __name__=="__main__":
    app = QApplication(sys.argv)
    myWindow = WindowClass()
    myWindow.show()
    app.exec_()
```



<from ~ import ~>

: from 뒤는 파일 이름. import 뒤에는 클래스, 변수

: ~파일에서 ~클래스를 불러오겠다. (*은 모든 함수와 클래스 의미)

: QMainWindow는 QtWidgets에서 불러온 클래스.

<class WindowClass(QMainWindow)>

: QMainWindow를 상속하는 WindowClass는 저희가 만들고 싶은 gui창을 만들어주는 클래스입니다.

<app = QApplication(sys.argv)>

: QApplication 클래스의 객체를 만들어 app이라고 부르기로 합니다.

Qapplication은 이 코드를 실제로 실행시키는 역할을 할 객체입니다.

<myWindow = WindowClass(>

: WindowClass클래스의 객체를 만들고 QMainWindow의 show() 메소드를 사용합니다. 실제로 창의 띄우는 역할을 하는 메소드)

<app.exec_(>

: exec_()는 프로그램을 이벤트 루프로 진입시키는 QApplication 클래스의 메소드

: 이벤트 루프는 프로그램을 무한 루프 안에서 계속 실행 시켜주어서 우리가 닫기 버튼을 누를 때 까지 창을 띄워주는 역할을 합니다.

과제 1 : PyQt5 창 띄우기

- week5-1.py 로 저장
- PyQt5로 기본 창을 띄우는 아래 코드를 따라 쳐보고 실행시켜서 확인해보세요!

```
import sys
from PyQt5.QtWidgets import *

class WindowClass(QMainWindow):
    def __init__(self):
        super().__init__()

if __name__=="__main__":
    app = QApplication(sys.argv)
    myWindow = WindowClass()
    myWindow.show()
    app.exec_()
```

과제 2 : 클래스의 상속

- week5-2.py 로 저장

<Vehicle 클래스>

- speed (속성)
: 시속 속도를 정수로 저장하는 속성
- mileage (속성)
: 운송수단이 사용되기 시작한 순간 부터 지금까지의 총 주행거리를 저장하는 속성(km 단위)
- daily_mileage_check (메소드)
: 오늘의 주행시간을 시 단위로 입력 받는다.
: speed 속성을 이용해 오늘의 주행거리를 계산하여 알려준다.(오늘의 주행거리=speed*오늘의 주행시간)
: 오늘의 주행거리를 mileage 속성에 더하여 총 주행거리를 알려준다.

<Bus 클래스>

- max_people (속성)
: 최대 탑승 인원을 저장하는 속성
- passengers (속성)
: 현재 탑승 인원을 저장하는 속성(생성자 안에서 0으로 초기화시켜 줍니다. 다음 슬라이드 참고)
- ride_on(메소드)
: 탑승할 인원을 매개변수로 받는다.
: 1명이 탑승할 때마다 '뽁!'을 출력하고 passengers를 증가시킨다. passengers가 max_people를 넘으면 더 이상 좌석이 남지 않았음을 알려주고 함수를 종료합니다. 함수를 종료하기 전에는 잔여 좌석 수를 알려줍니다.
- get_off(메소드)
: 내릴 인원을 매개변수로 받고, 잔여 좌석 수를 알려줍니다.

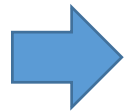
과제 2 실행 예시

Vehicle 클래스와 Bus 클래스의 생성자만 예시로 보여 드릴게요! 참고해서 과제 완성해주세요.

```
class Vehicle:
    def __init__(self, speed, mileage):
        self.speed = speed
        self.mileage = mileage
```

```
class Bus(Vehicle):
    def __init__(self, speed, mileage, max_people):
        super().__init__(speed, mileage)
        self.max_people = max_people
        self.passengers = 0
```

```
bus1 = Bus(50, 1800, 30)
bus1.ride_on(18)
bus1.ride_on(20)
bus1.get_off(10)
bus1.get_off(2)
bus1.daily_mileage_check()
```



삐! 삐! 삐! 삐! 삐! 삐! 삐! 삐! 삐! 삐! 삐! 삐! 삐! 삐! 삐! 삐! 삐!
잔여 12석

삐! 삐! 삐! 삐! 삐! 삐! 삐! 삐! 삐! 삐! 삐! 삐! 삐! 삐! 삐! 삐!
더 이상 좌석이 없습니다. 8명은 다음 버스를 이용해주세요.
잔여 0석

10명이 하차하였습니다.
잔여 10석

2명이 하차하였습니다.
잔여 12석

오늘의 주행 시간(시 단위) : 5

오늘의 주행거리 : 250km
총 주행거리 : 2050km

과제 제출 안내

과제 제출은 다음 주 월요일 자정 까지 입니다!

제출 이메일 : ektmf7890@hanyang.ac.kr

PyQt5 설치 안내

지금부터 PyQt5와 QtDesigner를 함께 설치해 보겠습니다.

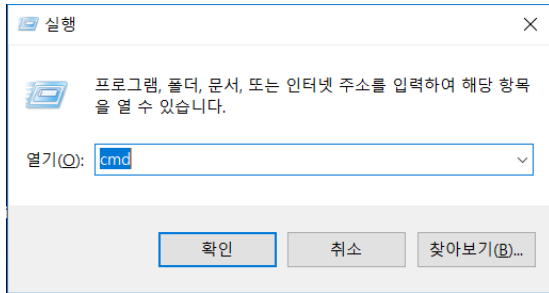
윈도우 사용자 이름이 한글로 되어있거나, 환경변수 설정이 되어있지 않는 등
다양한 이유로 설치 오류가 발생할 수 있습니다.

그런 오류를 같이 해결해드리기 위해서 수업 시간에 설치를 하는 것이기 때문에
설치에 문제가 생기시는 분들은 바로 채팅 창에 말씀해주시거나 저에게 카톡해주세요.

PyQt5 설치 안내

1. 명령 프롬프트 실행

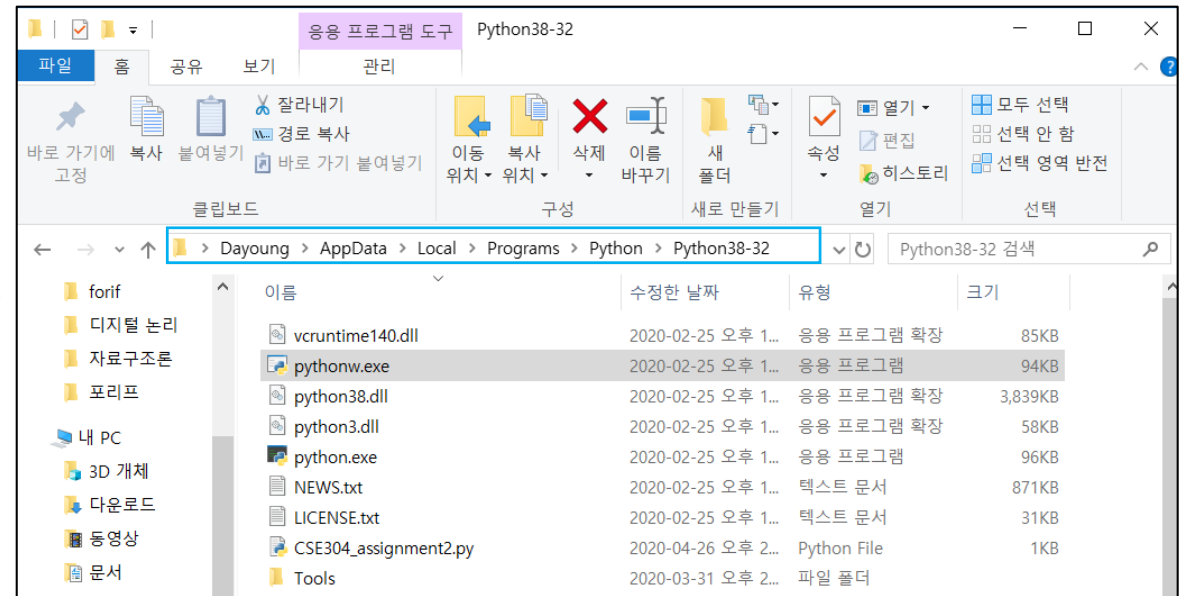
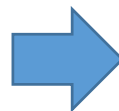
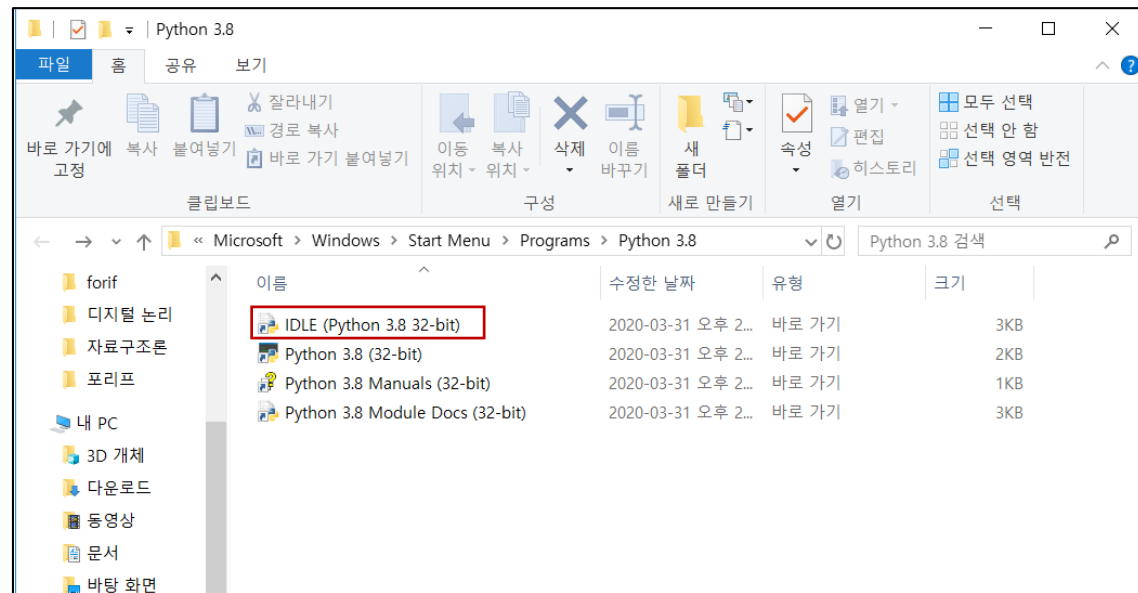
: '윈도우키+R' 누르면 뜨는 아래 창에 'cmd' 입력하고 실행



2. 파이썬 설치 폴더로 경로 확인

: idle 검색하신 후, 우 클릭해서 '파일 위치 열기'

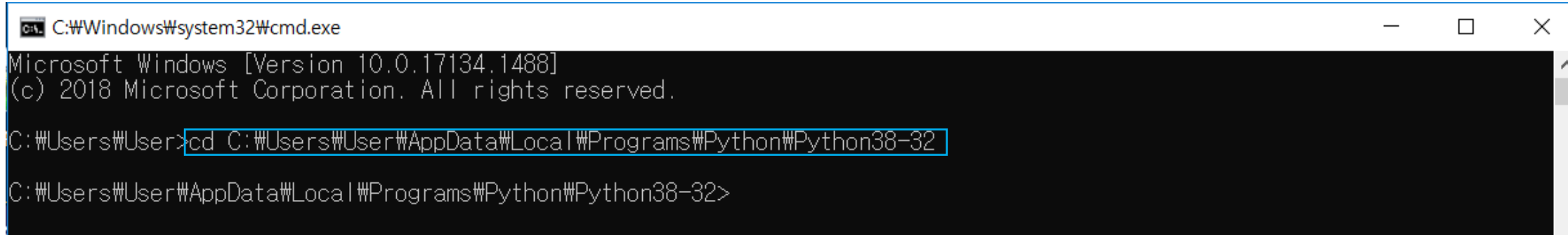
: 그 파일 안에서 IDLE 우 클릭하고 다시 '파일 위치 열기' -> 하늘색 박스 부분을 더블 클릭해서 파일 경로를 확인하세요.



PyQt5 설치 안내

3. 파이썬 설치 경로로 이동

cd C:\Users\Dayoung(사용자이름)\AppData\Local\Programs\Python\파이썬 버전명

A screenshot of a Windows Command Prompt window. The title bar shows 'C:\Windows\system32\cmd.exe'. The window content displays the Microsoft Windows version (10.0.17134.1488) and copyright information. The command 'cd C:\Users\User\AppData\Local\Programs\Python\Python38-32' is entered and highlighted with a blue selection box. The prompt shows the current directory as 'C:\Users\User\AppData\Local\Programs\Python\Python38-32>'.

```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.17134.1488]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\User>cd C:\Users\User\AppData\Local\Programs\Python\Python38-32
C:\Users\User\AppData\Local\Programs\Python\Python38-32>
```


PyQt5 설치 안내

4. PyQt5 설치

>pip install pyqt5

```
C:\Users\User\AppData\Local\Programs\Python\Python38-32>pip install pyqt5
```

5. Qt Designer 설치

>pip install pyqt5- tools

```
C:\Users\User\AppData\Local\Programs\Python\Python38-32>pip install pyqt5-tools
```

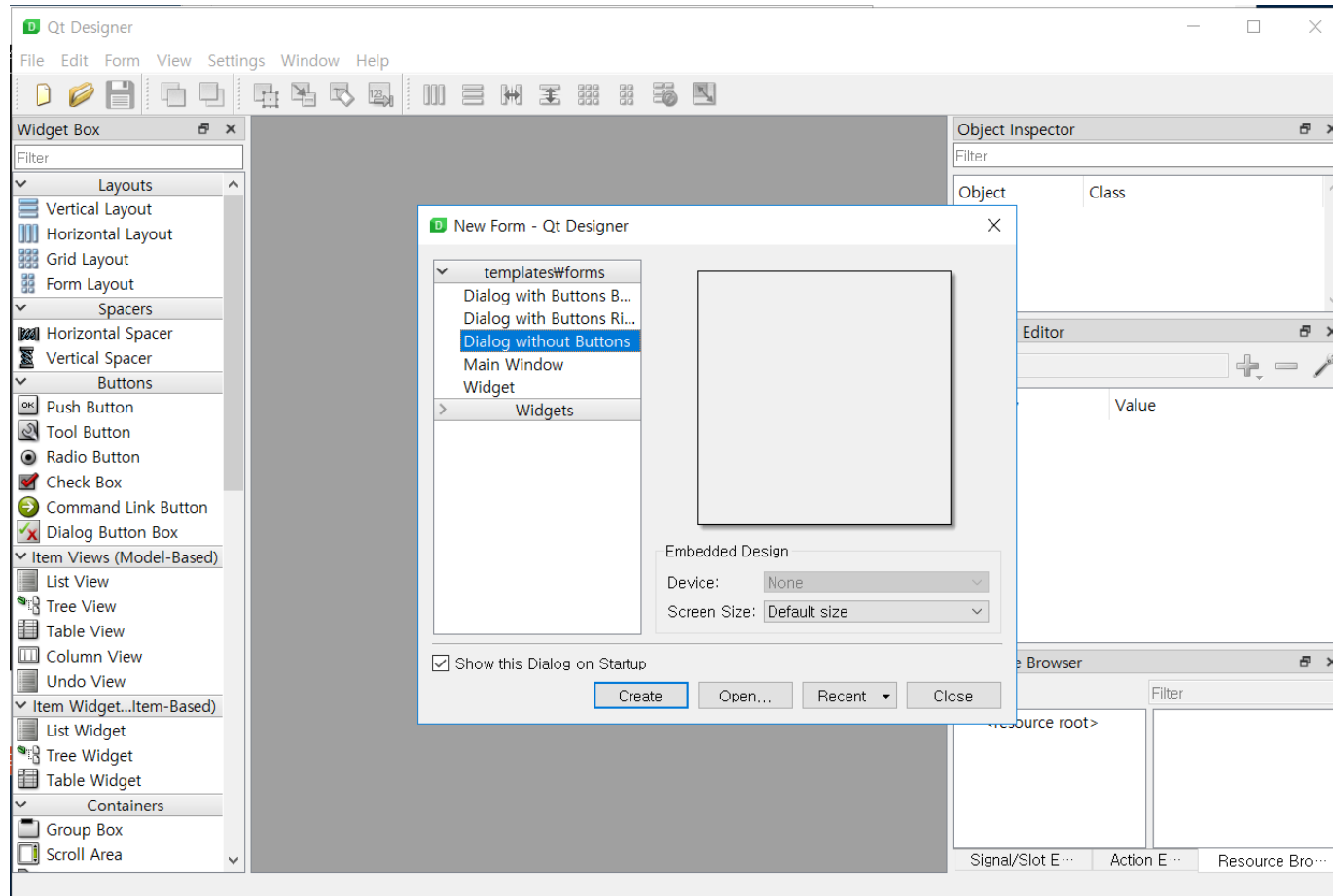
PyQt5 설치 안내

<Qt Designer 실행>

: 설치를 다 완료하셨으면 Qt Designer를 실행시켜 봅시다.

: 명령 프롬프트 창에 designer 명령어를 입력하고 조금 기다리면 아래 화면과 같이 실행.

```
C:\Users\User>designer
```



PyQt5 설치 안내

QtDesigner 실행까지 완료하신 분들은
채팅창에 '다운로드 정상적으로 완료!'라고 메시지 하나만
남겨주시고 화상 강의 나가시면 됩니당!

PyQt에 대해 먼저 공부해보실 수 있도록 참고 사이트 하나 첨부합니다.
과제 마무리하고 시간 있으시면 한 번 훑어보고 와주세요!

<https://wikidocs.net/35478>