파일 디렉터리 제어

파이썬 파일/디렉터리 제어!!!

- 텍스트 파일(txt, csv, log, 문서-xlsx, pptx, hwp...)
- 모니터링 (신규파일, 수정파일...) : 특정 디렉터리에 의심되는 파일?!
- 파일내에 중요 정보 포함?!!! -> 알람!!!, 모니터링!!

IT보안분야

- 해당 사이트 HTML, JS 등을 크롤링을 해서
- 스크립트(소스)내 #주석처리에 계정정보(테스트계정, IP정보)
  #, //, <!--
- 소스내에 이메일 정보(테스트이메일, 중요이메일)
- 특정 디렉터리에 새로운 파일들이 만들어진 것 확인!!!

파일과 파일 경로 정의

- 변수는 프로그램을 실행할 때 데이터를 저장하기 좋은 방법
- 그러나 프로그램 종료 후에도 데이터를 유지하고 싶다면, 파일 형태로 저장 필요
- 파이썬으로 하드 드라이브에 파일을 생성하고 읽고 쓰는 방법에 대해 다룰 것
- 파이썬의 `os` 모듈을 사용하여 파일 및 디렉터리를 제어할 것

파일과 파일 경로

- 파일에는 파일 이름과 경로라는 두 가지 주요 속성이 존재
- 경로는 컴퓨터에서 파일의 위치를 특정

```
# C:\Users\Pentest 경로에 # SK_shieldus_Rookies.txt 이름 파일 존재 # .txt는 파일 확장자라고 하며 파일 유형을 나타냄 C:\Users\Pentest\SK_shieldus_Rookies.txt
```

루트 폴더

- 루트 폴더는 모든 폴더의 최상위 폴더로 모든 폴더가 이 안에 포함
- 윈도우에서 루트 폴더는 `C:\` 이며 `C:` 드라이브 라고 부름
- macOS 또는 Linux의 루트 폴더는 `/`

```
C:\Users\Pentest\SK_shieldus_Rookies.txt
```

```
/usr/home/kali
```

현재 작업 중인 디렉터리 확인

- `os.getcwd()` 함수는 현재 작업 중인 디렉터리의 절대 경로를 문자열로 반환
- "현재 작업 디렉터리"는 현재 파이썬 스크립트가 실행되고 있는 폴더를 의미

```
import os current_directory = os.getcwd() print("Current working directory:", current_directory)
```

지정된 경로가 파일인지 폴더인지 판단

- `os.path.isfile(path)` 함수는 지정된 경로가 파일인 경우 `True` 를 반환

- 이 함수는 경로가 실제 파일을 가리키고, 그 파일이 존재하는 경우에만 `True`를 반환하므로 파일의 존재 여부와 유형을 확인할 때 유용

```
import os file_path = "example.txt" if os.path.isfile(file_path): print(f"
{file_path} is a file") else: print(f"{file_path} is not a file")
```

- `os.path.isdir(path)` 함수는 지정된 경로가 디렉터리인 경우 `True`를 반환
- 이 함수는 경로가 디렉터리를 가리키고, 해당 디렉터리가 존재하는 경우에만 `True`를 반환하므로, 디렉터리의 존재 여부를 확인하거나 경로가 디렉터리인지 확인할 때 사용

```
import os directory_path = "example_folder" if os.path.isdir(directory_pat
h): print(f"{directory_path} is a directory") else: print(f"{directory_pat
h} is not a directory")
```

`r""`로 묶인 문자열을 **"raw string"** 또는 "raw string literal"이라고 함

이것은 문자열 내부의 백슬래시(`\`)를 이스케이프 문자로 처리하지 않고, 있는 그대로 문자열에 포함시키기 위한 방법. 예를 들어, 일반 문자열에서는 `\n`이 줄바꿈을 의미하지만, raw string에서는 그대로 `\n`으로 해석. 파일 경로나 정규식처럼 백슬래시가 자주 사용되는 문자열을 다룰 때 유용

지정된 디렉터리 내 모든 파일과 디렉터리 목록 조회

- `os.listdir()` 함수는 지정된 경로의 디렉터리 내 모든 파일과 디렉터리 목록을 리스트 형태로 반환
- 경로를 지정하지 않으면 현재 디렉터리를 기준

```
import os entries = os.listdir('.') print("Entries in current directory:", entries)
```

```
import os entries = os.listdir('../') print("Entries in current director y:", entries)
```

디렉터리 트리 탐색

- `os.walk()` 메서드는 디렉터리 트리를 위에서 아래로 순회하며 각 디렉터리에 대한 정보를 제공
- 이 함수는 특히 디렉터리 구조를 재귀적으로 탐색할 때 매우 유용하며, 파일 시스템의 모든 파일과 디렉터리에 접근할 필요가 있을 때 자주 사용

`os.walk()` 메서드 사용 방법

- `os.walk()`는 시작 디렉터리를 인자로 받고, 디렉터리의 경로, 하위 디렉터리 목록, 그리고 파일 목록을 포함하는 튜플을 생성자 형태로 반환
- 반환 값은 (dirpath, dirnames, filenames)의 형태로, 각 요소는 다음을 의미
  - `dirpath`: 현재 디렉터리의 경로
  - `dirnames`: `dirpath` 아래에 있는 서브디렉터리의 리스트
  - `filenames`: `dirpath` 아래에 있는 파일의 리스트

```
# 코드 작성 import os for dirpath, dirnames, filenames in os.walk(r"C:\python_ex"): print(f"Found directory: {dirpath}") print(f"Subdir: {dirnames}") print(f"File: {filenames}") print("-"*50)
```

목표는 uploads 디렉터리 내의 txt 파일 확장자 파일만 가져온다.

```
import os dir_path = "uploads" all_files = os.listdir(dir_path) txt_files = [] for file in all_files: if file.endswith(".txt"): txt_files.append(file) print(txt_files)
```

파일 쓰기와 읽기

- 파일 객체를 사용하여 데이터를 읽는 방법
- 파일 읽기 테스트를 위한 예제 파일 생성

```
helloFile = open('example.txt', 'w', encoding='utf-8') helloFile.write('Line 1: Welcome to Python file handling.\n') helloFile.write('Line 2: This is the second line.\n') helloFile.write('Line 3: Here is the third line.\n') helloFile.write('Line 4: The file ends here.\n') helloFile.close()
```

with로 수정한 사례

```
# 파일에 텍스트 쓰기 with open('example.txt', 'w', encoding='utf-8') as file: file.write('안녕하세요, 파이썬입니다!\n') file.write('with 문을 사용하면 파일이 자동으로 닫힙니다.')
```

read()

- `read()` 메소드는 파일의 내용을 전부 읽어 하나의 문자열로 반환

```
helloFile = open('example.txt', 'r', encoding='utf-8') content = helloFile.read() print(content) helloFile.close()
```

- with 문 사용

```
# 'with' 문을 사용하여 파일을 열고 자동으로 닫기 with open('example.txt', 'r', encoding='utf-8') as helloFile: content = helloFile.read() print(content)
```

```
Line 1: Welcome to Python file handling. Line 2: This is the second line. Line 3: Here is the third line. Line 4: The file ends here.
```

readline()

- `readline()` 메소드는 파일에서 한 줄을 읽고 문자열로 반환
- 한 줄씩 읽어오며, 호출할 때마다 다음 줄을 읽습니다.

```
helloFile = open('example.txt', 'r', encoding='utf-8') print(helloFile.readline()) # 첫 번째 줄 읽기 print(helloFile.readline()) # 두 번째 줄 읽기 helloFile.close()
```

```
Line 1: Welcome to Python file handling. Line 2: This is the second line.
```

readlines()

- `readlines()` 메소드는 파일의 모든 줄을 읽어 각 줄을 문자열로 갖는 `리스트 반환`
- 모든 줄을 읽어서 리스트로 반환한 다음, 이 리스트를 그대로 출력

```
helloFile = open('example.txt', 'r', encoding='utf-8') lines = helloFile.readlines() print(lines) helloFile.close()
```

```
['Line 1: Welcome to Python file handling.\n', 'Line 2: This is the second line.\n', 'Line 3: Here is the third line.\n', 'Line 4: The file ends her e.\n']
```

```
with open('example.txt', 'r', encoding='utf-8') as helloFile: content = helloFile.readlines() for line in content: print(line.strip())
```

```
with open('example.txt', 'r', encoding='utf-8') as hellofile: content = hellofile.readlines() for line in content: print(line, end='')
```

```
# 'with' 문을 사용하여 파일을 열고 자동으로 닫기 with open('example.txt', 'r', encoding='utf-8') as helloFile: content = helloFile.read() print(content) print("======================") with open('example.txt', 'r', encoding='utf-8') as helloFile: content = helloFile.readline() print(content) print("======================") with open('example.txt', 'r', encoding='utf-8') as helloFile: content = helloFile.readlines() print(content)
```

readline()을 이용한 파일 읽기 예제

```
#uploads 폴더의 디렉터리 정보를 수집 #파일 확장자가 txt 인 것을 리스트화 #txt 파일을 열어서 출력 import os dir_path = "uploads" all_files = os.listdir(dir_path) txt_files = [] for file in all_files: if file.endswith(".txt"): txt_files.append(file) # 1.txt, 2.txt /uploads/1.txt for filename in txt_files: file_path = os.path.join(dir_path, filename) with open(file_path, 'r', encoding='utf-8') as file: print(f"{filename} 내용:") print(file.read()) print("-" * 40)
```

파일 모니터링 기능!!!

```
import os import time from datetime import datetime dir_path = "uploads" pre_file = set(os.listdir(dir_path)) while True: #날짜 표시 now = datetime.now() day = now.strftime("%Y-%m-%d") hour = now.strftime("%H:%M:%S") current_file = set(os.listdir(dir_path)) result_diff = current_file - pre_file for file_name in result_diff: print(f"새로운 파일 탐지: {file_name}") with open(f"{day}_탐지 보고서.txt", "a", encoding="UTF-8") as file: file.write(f"작성자 : 조정원\n") file.write(f"주요 내용: 신규 파일 탐지\n") file.write(f"시간 {hour}, 파일 이름: {file_name}\n") file.write(f"======================") pre_file = current_file print("모니터링 중") time.sleep(1)
```

소스코드 내에 중요한 정보가 포함되어 있는지 확인?!!!

- 주석처리 내용 내 테스트 계정 정보 포함 여부?!
- 주석처리 내용 내 내부 IP 정보들 포함 여부
- 소스코드 내에 개인정보(이메일, 주민번호 등)

개발 서버 (Close) → 스테이지 서버(Stage Server, Close) → 오픈 서버(대외 서버, Open)

```
#uploads 폴더의 디렉터리 정보를 수집 #파일 확장자가 txt 인 것을 리스트화 #txt 파일을 열어서 출력 import os dir_path = "uploads" all_files = os.listdir(dir_path) txt_files = [] for file in all_files: if
```