Python을 이용한

유니코드 아트 생성기

201704044 백문하

아스키 아트 & 유니코드 아트란?

- 아스키 아트 (ASCII Art)
  7비트의 문자 코드인 ASCII 코드 문자들을 이용하여 그림을 그림
  눈에 보이는 0x20부터 0x7F까지의 문자들을 이용

  문자로 그린 모든 그림을 아스키 아트라고 칭하기도 함

- 유니코드 아트 (Unicode Art)
  4바이트로 표현되는 다양한 유니코드 문자를 이용한 그림

  한글, 한자, 다양한 특수문자를 이용하려면
  유니코드를 사용해야 함

사용 환경

- Python
  리스트 이용이 간편하고 표준 라이브러리에 JSON 파서가 존재함

- Pillow
  파이썬 이미지 라이브러리(PIL)의 포크
  밝기 측정, 이미지 픽셀 값 계산 등

- imageio
  동영상 입출력 기능

생성 과정

1. 서체 밝기 측정 & 정규화

2. 사진 크기 조정

3. 사진 밝기 측정 & 정규화

4. 밝기 매칭

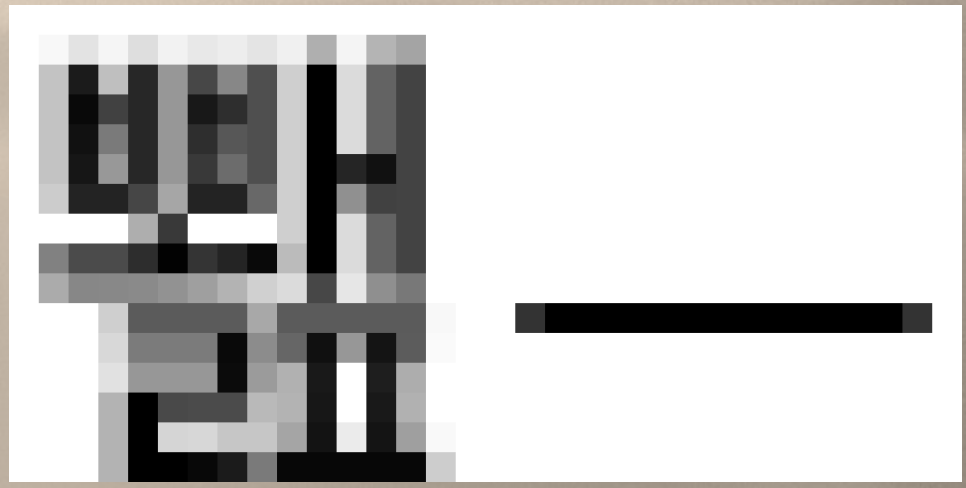
5. 변환 완료

서체 밝기 측정

사용할 서체의 문자 밝기를 측정
서체 크기만한 사진 img에 글자를 써서
사진 밝기의 평균값 mean을 계산
직접 계산할 필요 없음

```
stat = PIL.ImageStat.Stat(img)
mean = stat.mean[0]
```

이후 이진 탐색을 위해 반드시 정렬



서체 밝기 정규화

가장 어두운 글자는 검정색과 가깝지 않음
공백이 없다면, 가장 밝은 글자도 흰색과 가깝지 않음
나중에 이미지도 마찬가지로 정규화 해 줌

최대값 : max, 최소값 : min,
차이값 : dif, 픽셀 밝기 : avg,
최종 픽셀 밝기 : val

```
dif = max - min
val = ((avg - min) / dif) * 255
```

사진 크기 조정

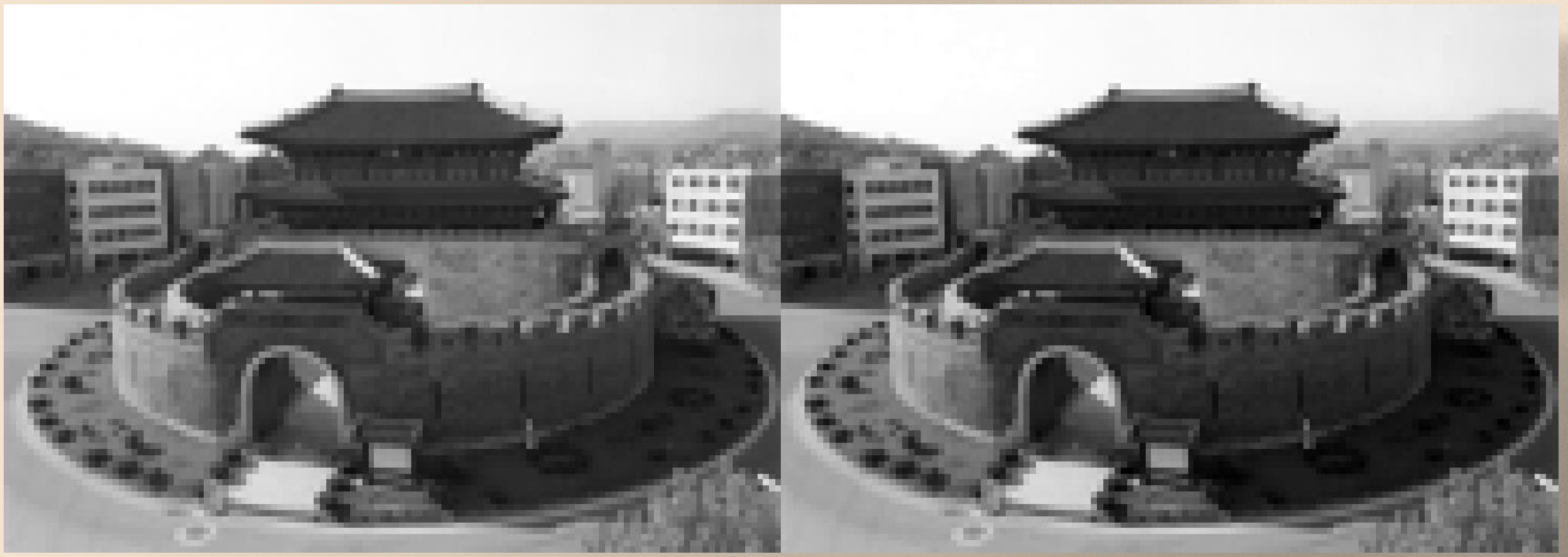
픽셀 수가 매우 많으므로, 사용할 때는 이미지를 미리 리사이징
가로 크기 기준, 세로 크기 기준, 비율 변경 가능하도록 (ASCII 지원)

사진 밝기 측정

변환할 사진 픽셀의 R, G, B 평균값이 밝기임

이미지 밝기 정규화

밝은 부분은 더 밝게, 어두운 부분은 더 어둡게 된 것을 볼 수 있음

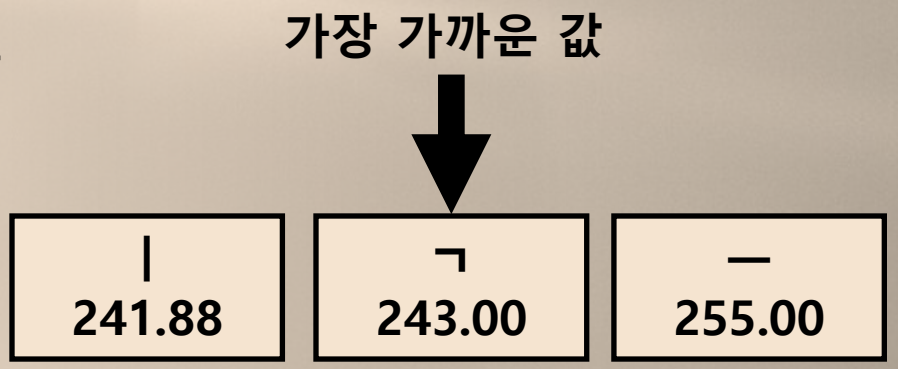


밝기 매칭

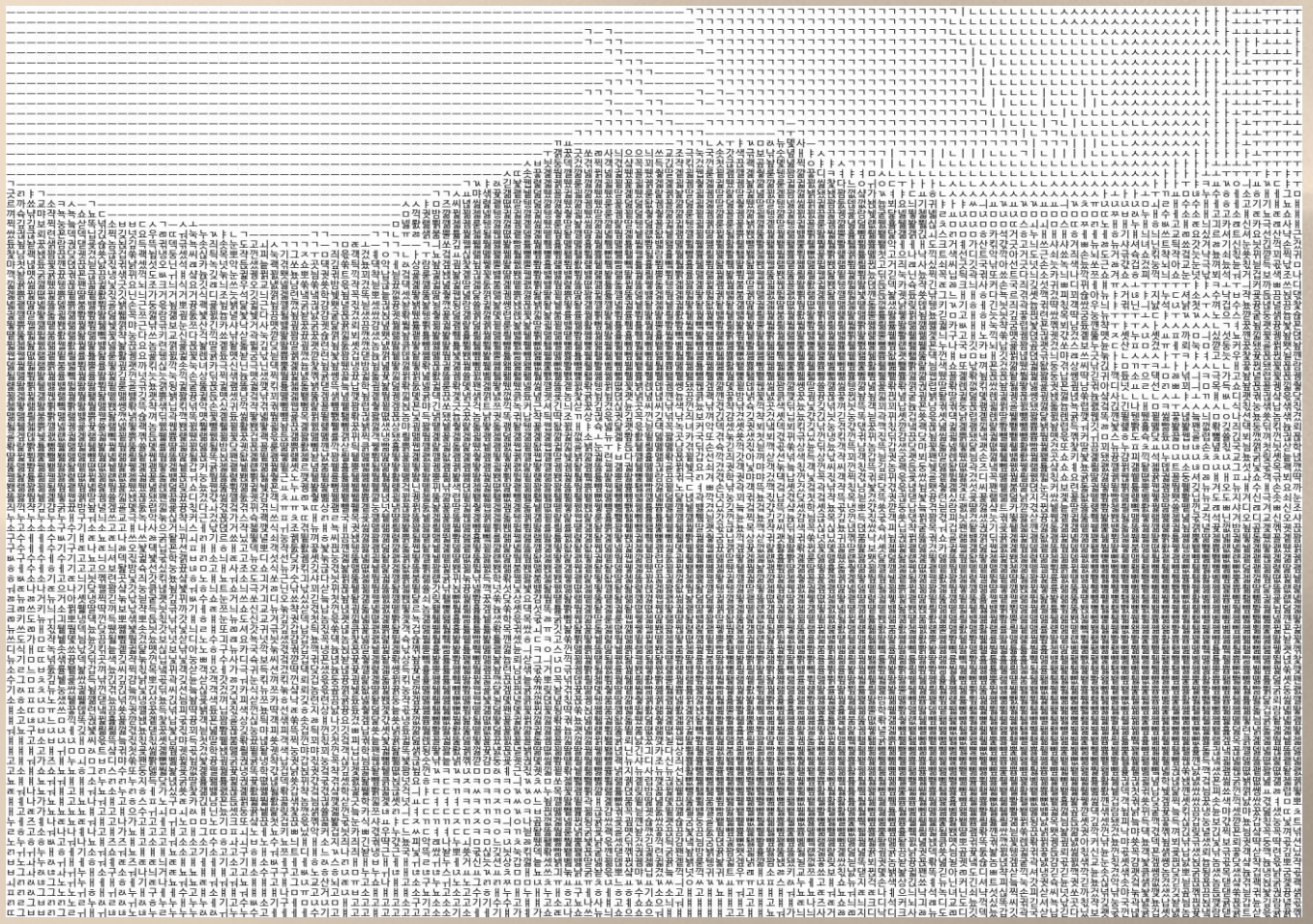
가장 가까운 값을 찾기 위해
정렬된 서체 밝기 데이터 리스트에서 이진 탐색을 이용

파이썬의 bisect.bisect()는 찾지 못하면 다음 인덱스를 반환
따라서 이전 인덱스와 비교, 가까운 값으로 변환

예) 밝기 244인 픽셀



결과물



서체와 사진 정규화를 하지 않는다면

오히려 밝기가 더 일치하는 모습을 보임
그러나 상의 구분이 힘들어짐

다른 사진

사진 변환 결과물

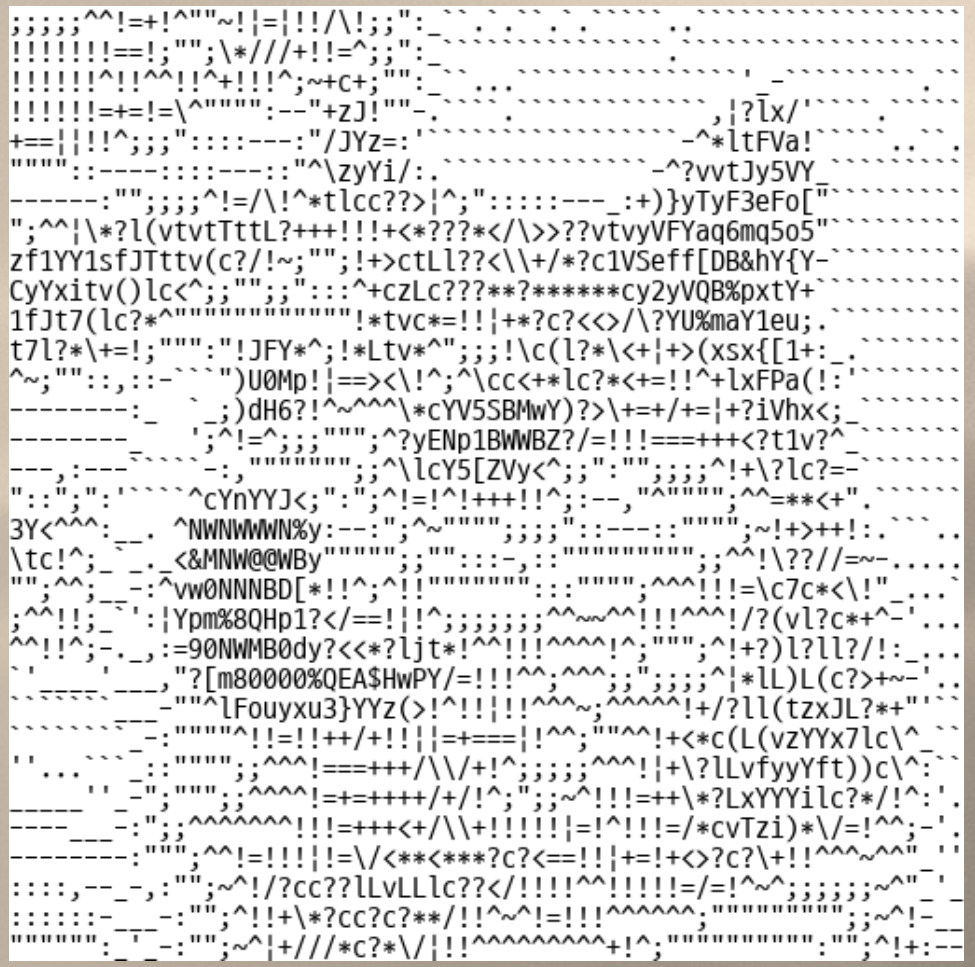
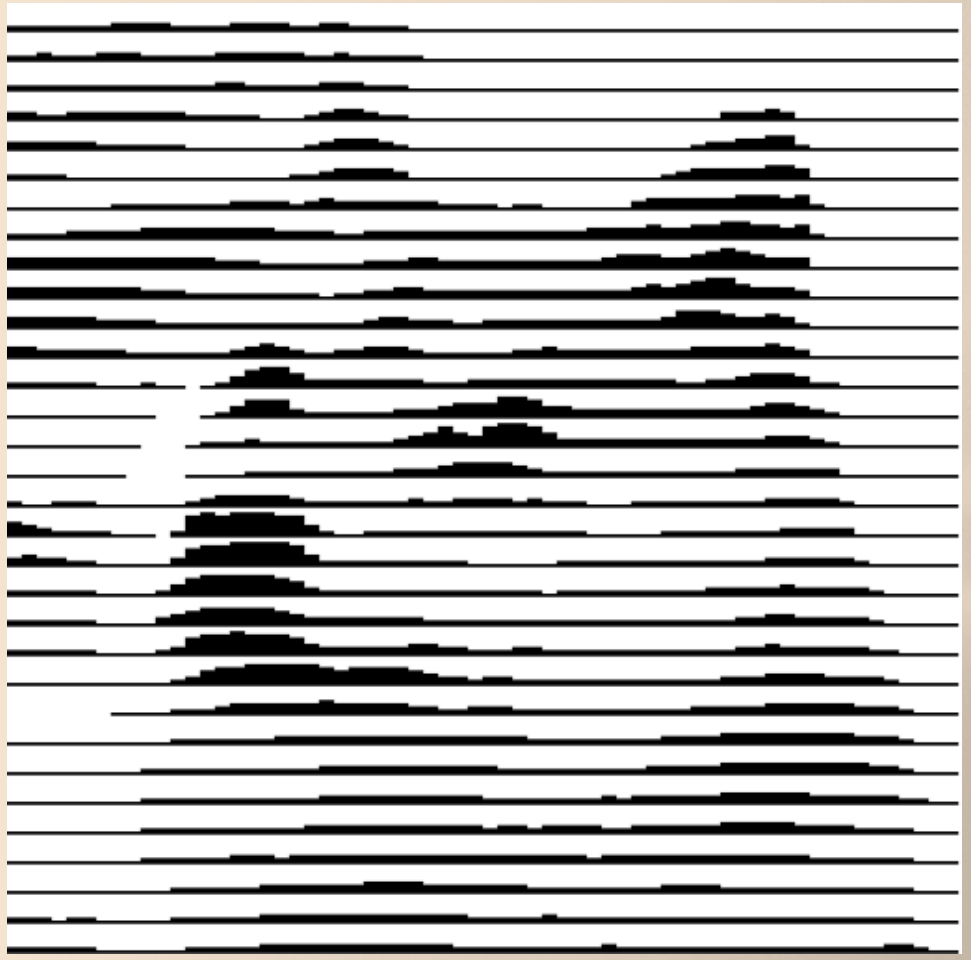


사진 변환 결과물



설정 파일의 활용

매번 프로그램을 실행할 때마다 사용할 글자와 서체
사진을 선택하는 것은 번거로움
따라서 JSON 설정 파일을 자동 생성 후 편집하여 활용

문자의 유니코드 범위
서체의 파일명
측정용 샘플의 서체 크기
정규화 여부

```
{
    "ranges": [
        [
            44032,
            55203
        ],
        [
            12593,
            12643
        ]
    ],
    "font_file_name": "fonts\\d2coding.ttf",
    "font_size": 8,
    "normalize": true
}
```

서체 밝기 데이터 보관

서체 밝기 측정은 상당히 오래 걸리는 작업
따라서 측정한 밝기와 해당하는 글자를 JSON으로 저장하여 재활용

가장 어두운 글자들
49030 뾆
49029 뾡
가장 밝은 글자들
12641 ㅡ
12593 ㄱ

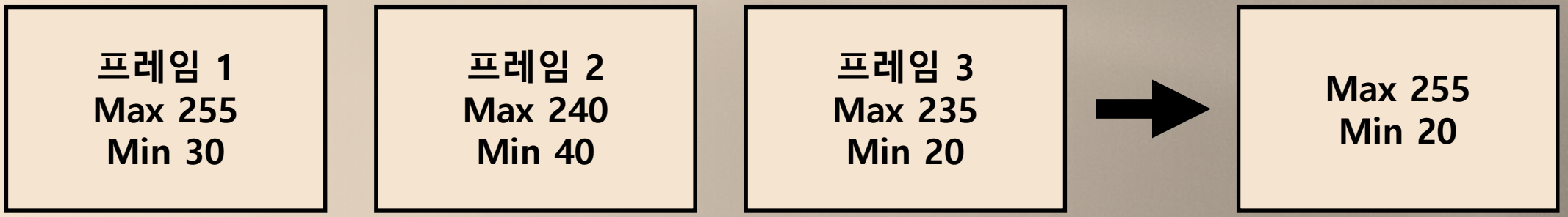

```
[
    [
        0.0,
        49030
    ],
    [
        2.056451612903226,
        49029
    ],
    [
        2.154377880184332,
        49031
    ],
```

```
    [
        241.8778801843318,
        12643
    ],
    [
        243.0040322580645,
        12593
    ],
    [
        255.0,
        12641
    ]
]
```

영상의 변환

영상에서 각 프레임을 얻어 변환한 후 다시 영상으로 묶음

영상 변환 시 각 사진별로 정규화를 실행하면 밝기가 들쑥날쑥함
따라서 영상의 모든 프레임을 분석하여 최대, 최소 밝기를 찾음



영상 예시 1 - 흑백 고대비

원본 : https://www.youtube.com/watch?v=i41KoE0iMYU

변환 : https://www.youtube.com/watch?v=jzUBBMx9jJ0

모듈화

추후에 라이브러리 혹은 GUI 프로그램으로 활용할 수 있도록
변환의 각 작업을 맡는 클래스로 구성

서체 설정 로드 => FontDataSettings
서체 설정의 정보로 서체 밝기를 측정 => FontData
서체 밝기와 로드한 이미지로 이미지 변환 => ImageData
변환 결과를 텍스트 혹은 이미지로 출력 => TextData
동영상 입출력 => VideoData

차후 개선점

1. GUI 제작
   JSON 이용은 생 코드보단 편리하지만,
   일반 사용자들에게는 불편할 것

   Qt(PySide6)를 이용한 GUI 제작 계획

2. 동영상 입출력 시 지나친 자원 낭비
   해상도가 작고 길이가 짧은 영상 조차도 변환 시
   시간을 수 분 잡아먹음

   과정을 병렬화하고, 만든 임시파일을 즉시 사용 후 제거

감사합니다