```
# Doc Number: Korea JTC1/SC2: k2653_282_word_to_png_transfer.py

# Date: 2026.06.11.
# Author: SHIN, SangHyun


import argparse
import os
import win32com.client
import fitz                          # pymupdf  — pip install pymupdf
from PIL import Image, ImageChops  # pillow — pip install pillow
# ── Word 상수
_____________________________________________________________________________
_____________________________________
WD_ACTIVE_END_PAGE_NUMBER = 3    # wdActiveEndPageNumber
WD_STAT_PAGES                = 4    # wdStatisticPages
WD_PAGE_PT_MAX               = 1584.0    # Word 허용 최대 페이지 크기(포인트)
WD_PAGE_PT_MIN               = 7.2       # Word 허용 최소 페이지 크기(포인트)
# ── 기본값 (커맨드라인 인자로 덮어씀)
_____________________________________________________________________________
DPI = 400
#
_____________________________________________________________________________
_____________________________________________________________
# 유틸리티
#
_____________________________________________________________________________
_____________________________________________________________
def autocrop_white(img: Image.Image, dpi: int) -> Image.Image:
    """
    흰색(255,255,255) 배경을 기준으로 이미지 여백 자동 제거 후,
    상하좌우에 정확히 1mm의 흰색 여백 추가.
    """
    bg    = Image.new("RGB", img.size, (255, 255, 255))
    diff = ImageChops.difference(img, bg)
    bbox = diff.getbbox()             # 비흰색 픽셀의 bounding box
    if not bbox:
        return img

    cropped = img.crop(bbox)

    # 1mm에 해당하는 픽셀 수 계산 (1 inch = 25.4 mm)
    margin_px = int(round(dpi * 1.0 / 25.4))

    new_w = cropped.width + 2 * margin_px
    new_h = cropped.height + 2 * margin_px

    # 흰색 배경에 크롭된 이미지를 중앙에 배치
```

```
    padded = Image.new("RGB", (new_w, new_h), (255, 255, 255))
    padded.paste(cropped, (margin_px, margin_px))
    return padded
def get_total_pages(doc) -> int:
    """문서 전체 페이지 수."""
    try:
        return doc.ComputeStatistics(WD_STAT_PAGES)
    except Exception:
        try:
            return doc.BuiltInDocumentProperties("Number of Pages").Value
        except Exception:
            return 1
def get_all_tables_by_page(doc) -> dict:
    """
    문서의 모든 테이블을 {페이지번호: [테이블1-based인덱스, ...]} 로 반환.
    테이블 첫 번째 셀의 페이지 기준.
    """
    page_table_map = {}
    for i in range(1, doc.Tables.Count + 1):
        try:
            page = doc.Tables(i).Cell(1,
1).Range.Information(WD_ACTIVE_END_PAGE_NUMBER)
            page_table_map.setdefault(page, []).append(i)
        except Exception as e:
            print(f"  [경고] 테이블 {i} 페이지 확인 실패: {e}")
    return page_table_map
#
──────────────────────────────────────────────────────────────────────────
────────────────────────────────────────────────────────────────

# 핵심 변환 함수
#
──────────────────────────────────────────────────────────────────────────
────────────────────────────────────────────────────────────────

def table_to_png(word, table, output_png: str, temp_pdf: str) -> tuple[int, int]:
    """
    Word 테이블 하나를 고해상도 PNG로 변환.
    전략:
    - 임시 문서 여백을 0으로 설정, 페이지 크기를 최대(1584pt)로 설정
      → 테이블이 가급적 적은 페이지로 분할됨
    - PDF 내보내기 후 PyMuPDF로 래스터화
    - 여러 페이지가 나오면 Pillow로 수직 스티칭
    - 흰 여백 자동 크롭 → 테이블 콘텐츠만 남은 PNG 저장
    반환: (width_px, height_px)
    """
    new_doc = word.Documents.Add()
    try:
        ps = new_doc.PageSetup
        # ① 원본 테이블의 편집용지 너비 읽기 (A4 ≈ 595pt, 가로 842pt 등)
```

```
        #     콘텐츠 너비 = 페이지 너비 - 좌우 여백  (여백은 새 문서에서 0으로
설정하므로 그 값을 사용)
        try:
            src_ps       = table.Range.PageSetup
            page_width   = src_ps.PageWidth                                  # 예: A4 =
595.3pt
            # 원본 여백을 제외한 실제 콘텐츠 너비
            content_w    = page_width - src_ps.LeftMargin - src_ps.RightMargin
            content_w    = max(min(content_w, WD_PAGE_PT_MAX),
WD_PAGE_PT_MIN)
        except Exception:
            content_w = 451.0    # fallback: A4 기본 여백 제거 시 콘텐츠 너비
(~15.9cm)
        # ② 여백을 먼저 0으로 설정 (PageHeight 설정 전 "여백 > 페이지" 오류
방지)
        for attr in ("TopMargin", "BottomMargin", "LeftMargin", "RightMargin"):
            try:
                setattr(ps, attr, 0)
            except Exception:
                setattr(ps, attr, WD_PAGE_PT_MIN)
        # ③ 페이지 너비 = 원본 콘텐츠 너비, 높이 = 최대 (페이지 분할 최소화)
        ps.PageWidth  = content_w
        ps.PageHeight = WD_PAGE_PT_MAX
        # ④ 테이블 복사 → 새 문서에 붙여넣기 (원본 서식 유지를 위해
PasteAndFormat 사용)
        table.Range.Copy()
        new_doc.Range(0, 0).PasteAndFormat(16)  # 16 =
wdFormatOriginalFormatting
        # ⑤ 16열 한자 표인 경우 열 너비 균일화 및 줄바꿈 방지 설정
        pasted_table = new_doc.Tables(1)
        if pasted_table.Columns.Count == 16:
            try:
                pasted_table.AllowAutoFit = False
                # ① 열 너비 균일화 설정
                for c in range(1, 17):
                    if c % 2 != 0:
                        # 번호 열 (홀수 열): 조금 넓게 설정 (32.5 / 468.0 비율)
                        pasted_table.Columns(c).Width = content_w * (32.5 /
468.0)
                    else:
                        # 한자 열 (짝수 열): 조금 좁게 설정 (26.0 / 468.0 비율)
                        pasted_table.Columns(c).Width = content_w * (26.0 /
468.0)

                # ② 단락 공백 제거 및 줄바꿈 규칙 강제 적용 (행간 어긋남 방지)
                pasted_table.Range.ParagraphFormat.SpaceBefore = 0.0
                pasted_table.Range.ParagraphFormat.SpaceAfter = 0.0
                pasted_table.Range.ParagraphFormat.LineSpacingRule = 0  #
```

```
wdLineSpaceSingle (단일 행간)

                # ③ 모든 셀의 상하 수직 정렬을 가운데 정렬로 설정
(wdCellAlignVerticalCenter = 1)
                pasted_table.Range.Cells.VerticalAlignment = 1

                # ④ 행 높이(행간) 일치화 설정 (wdRowHeightExactly = 2)
                pasted_table.Rows(1).HeightRule = 2
                pasted_table.Rows(1).Height = 16.0
                for r in range(2, pasted_table.Rows.Count + 1):
                    pasted_table.Rows(r).HeightRule = 2
                    pasted_table.Rows(r).Height = 30.5
                # ⑤ 테두리 설정: 모든 테두리 제거 (음영이 연속적인 띠가 되도록
설정)
                for b_idx in range(-8, 1):
                    try:
                        pasted_table.Borders(b_idx).LineStyle = 0
                    except Exception:
                        pass
            except Exception as e:
                print(f"       [경고] 표 규격(열 너비/행 높이/정렬) 조정 실패 (병합된
셀이 있을 수 있음): {e}")
        # ⑥ PDF 내보내기
        new_doc.ExportAsFixedFormat(
            OutputFileName       = os.path.abspath(temp_pdf),
            ExportFormat         = 17,     # wdExportFormatPDF
            OpenAfterExport      = False,
            OptimizeFor          = 0,      # wdExportOptimizeForPrint (최고 화질)
            BitmapMissingFonts = True,
            UseISO19005_1        = False,
        )
    finally:
        new_doc.Close(False)
    # ⑤ PDF → 이미지 래스터화 (모든 페이지)
    mat      = fitz.Matrix(DPI / 72, DPI / 72)   # 72dpi 기준 배율 변환
    pdf_doc = fitz.open(temp_pdf)
    page_imgs = []
    for pi in range(len(pdf_doc)):
        pix = pdf_doc[pi].get_pixmap(matrix=mat, alpha=False)
        page_imgs.append(Image.frombytes("RGB", [pix.width, pix.height],
pix.samples))
    pdf_doc.close()
    n_pages = len(page_imgs)
    # ⑥ 수직 스티칭 (단일 페이지면 그대로)
    if n_pages == 1:
        stitched = page_imgs[0]
    else:
        total_w = max(im.width  for im in page_imgs)
```

```
        total_h = sum(im.height for im in page_imgs)
        stitched = Image.new("RGB", (total_w, total_h), (255, 255, 255))
        y_off = 0
        for im in page_imgs:
            stitched.paste(im, (0, y_off))
            y_off += im.height
    # ⑦ 흰 여백 자동 크롭 (테이블 외 여백 제거) 및 1mm 여백 추가
    cropped = autocrop_white(stitched, DPI)
    # ⑧ PNG 저장
    cropped.save(output_png, "PNG")
    # 임시 PDF 삭제
    try:
        os.remove(temp_pdf)
    except Exception:
        pass
    suffix = f", {n_pages}장 스티칭" if n_pages > 1 else ""
    print(f"        ✓ {os.path.basename(output_png)}"
          f"  ({cropped.width}×{cropped.height}px, {DPI}dpi{suffix})")
    return cropped.width, cropped.height
#
─────────────────────────────────────────────────────────────────────────────
─────────────────────────────────────────────────────────
# 메인 처리
#
─────────────────────────────────────────────────────────────────────────────
─────────────────────────────────────────────────────────
def export_all_tables(docx_path: str):
    """지정한 Word 파일의 모든 테이블을 고해상도 PNG로 변환."""
    # 출력 폴더: Word 파일 옆  output_tables/<파일명>/  자동 생성
    doc_dir    = os.path.dirname(os.path.abspath(docx_path))
    doc_stem   = os.path.splitext(os.path.basename(docx_path))[0]
    output_dir = os.path.join(doc_dir, "output_tables", doc_stem)
    os.makedirs(output_dir, exist_ok=True)
    print(f"파일  : {os.path.basename(docx_path)}")
    print(f"출력  : {output_dir}")
    word = win32com.client.Dispatch("Word.Application")
    word.Visible = False
    doc = word.Documents.Open(os.path.abspath(docx_path))
    saved = []
    try:
        total_pages  = get_total_pages(doc)
        total_tables = doc.Tables.Count
        print(f"페이지: {total_pages}쪽 | 테이블: {total_tables}개\n")
        if total_tables == 0:
            print("테이블이 없습니다.")
            return []
        page_table_map = get_all_tables_by_page(doc)
        for page_num in sorted(page_table_map.keys()):
```

```
            indices = page_table_map[page_num]
            print(f"  [페이지 {page_num}] 테이블 {len(indices)}개")
            for seq, tbl_idx in enumerate(indices, start=1):
                print(f"    → 테이블 {tbl_idx} 변환 중...")
                table    = doc.Tables(tbl_idx)
                out_png  = os.path.join(output_dir,
f"page{page_num:03d}_table{seq:02d}.png")
                temp_pdf = os.path.join(output_dir,
f"_tmp_{page_num:03d}_{seq:02d}.pdf")
                try:
                    table_to_png(word, table, out_png, temp_pdf)
                    saved.append(out_png)
                except Exception as e:
                    print(f"       [오류] {e}")
    finally:
        doc.Close(False)
        word.Quit()
    print(f"\n완료! PNG {len(saved)}개 → {output_dir}")
    return saved
#
─────────────────────────────────────────────────────────────────────────
─────────────────────────────────────────────────────────
# 진입점
#
─────────────────────────────────────────────────────────────────────────
─────────────────────────────────────────────────────────
if __name__ == "__main__":
    import sys
    try:
        sys.stdout.reconfigure(encoding='utf-8')
        sys.stderr.reconfigure(encoding='utf-8')
    except Exception:
        pass
    parser = argparse.ArgumentParser(
        description="Word 문서의 모든 테이블을 고해상도 PNG로 변환합니다."
    )
    parser.add_argument(
        "docx",
        metavar="Word파일경로",
        help="변환할 .docx / .doc 파일의 경로"
    )
    parser.add_argument(
        "--dpi",
        type=int,
        default=400,
        metavar="N",
        help="출력 해상도 dpi (기본값: 400)"
    )
```

```
    args = parser.parse_args()
    # 파일 탐색 및 유효성 검사 (CWD 및 스크립트 디렉토리 탐색, 확장자 보정)
    input_path = args.docx
    resolved_path = None
    search_dirs = [os.getcwd(), os.path.dirname(os.path.abspath(__file__))]
    extensions = ["", ".docx", ".doc"]
    for s_dir in search_dirs:
        for ext in extensions:
            # 입력한 파일명에 이미 확장자가 있는데 다른 확장자를 덧붙이는 것 방지
            if ext != "" and input_path.lower().endswith((".docx", ".doc")):
                continue
            candidate = os.path.join(s_dir, input_path + ext)
            if os.path.isfile(candidate):
                resolved_path = os.path.abspath(candidate)
                break
        if resolved_path:
            break
    if not resolved_path:
        print(f"[오류] 파일을 찾을 수 없습니다: {input_path}")
        print("  → 현재 작업 디렉토리(CWD)와 스크립트 디렉토리를 모두
검색했습니다.")
        print('  → 파일명에 공백이 있으면 큰따옴표로 감싸세요.')
        print(f'  예) python word_to_png_transfer.py "파일명 포함 공백.docx"')
        raise SystemExit(1)
    if not resolved_path.lower().endswith((".docx", ".doc")):
        print(f"[오류] Word 파일(.docx/.doc)이 아닙니다: {resolved_path}")
        raise SystemExit(1)
    docx_path = resolved_path
    DPI = args.dpi
    export_all_tables(docx_path)
```