

# Escalar contenido PDF para impresión

Necesito imprimir documentos y he notado que el espacio en blanco alrededor del contenido suele ser demasiado grande, lo que desperdicia papel y hace que el texto parezca más pequeño de lo que debería. Este script ayuda a escalar automáticamente el contenido del PDF para que se ajuste mejor a la página, detectando el área del contenido y escalándola para llenar la página, respetando un pequeño margen.

```
import subprocess
import sys
import os
from PIL import Image
from pdf2image import convert_from_path

MARGIN_PERCENT = 0.005
DPI = 72

def convertir_pixeles_a_puntos(pixeles, dpi):
    """Convierte pixeles a puntos."""
    return pixeles * 72 / dpi

def obtener_dimensiones_imagen(imagen):
    """Obtiene las dimensiones de la imagen en pixeles y puntos."""
    ancho, alto = imagen.size
    dpi = imagen.info.get('dpi', (DPI, DPI))
    ancho_puntos = convertir_pixeles_a_puntos(ancho, dpi[0])
    alto_puntos = convertir_pixeles_a_puntos(alto, dpi[1])
    return ancho, alto, ancho_puntos, alto_puntos, dpi

def analizar_espacio_blanco(imagen, ancho, alto):
    """Analiza el espacio en blanco para encontrar el cuadro delimitador del contenido."""
    margen_izquierdo_px = ancho
    margen_derecho_px = 0
    margen_superior_px = alto
    margen_inferior_px = 0
    contenido_encontrado = False

    for x in range(ancho):
        for y in range(alto):
            pixel = imagen.getpixel((x, y))
            if isinstance(pixel, tuple):
```

```

        if any(c < 250 for c in pixel):

            if not contenido_encontrado:

                margen_izquierdo_px = x
                margen_superior_px = y
                contenido_encontrado = True

            margen_derecho_px = max(margen_derecho_px, x)
            margen_inferior_px = max(margen_inferior_px, y)

        elif pixel < 250:

            if not contenido_encontrado:

                margen_izquierdo_px = x
                margen_superior_px = y
                contenido_encontrado = True

            margen_derecho_px = max(margen_derecho_px, x)
            margen_inferior_px = max(margen_inferior_px, y)

    if not contenido_encontrado:

        return None, None, None, None

margen_derecho_px = ancho - margen_derecho_px
margen_inferior_px = alto - margen_inferior_px
return margen_izquierdo_px, margen_derecho_px, margen_superior_px, margen_inferior_px

def calcular_factor_escala(input_pdf):
    """
    Detecta las dimensiones de la primera página de un PDF, analiza el espacio en blanco,
    y calcula el factor de escala basado en el contenido del PDF y las dimensiones objetivo A4 con márgenes.
    Devuelve el factor de escala o None si ocurre un error.
    """

    print(f"Calculando factor de escala para: {input_pdf}")

    try:
        imagenes = convert_from_path(input_pdf, first_page=1, last_page=1)
        if not imagenes:
            print("No se pudo convertir el PDF a imagen.")
            return None

        imagen = imagenes[0]
        ancho, alto, ancho_puntos, alto_puntos, dpi = obtener_dimensiones_imagen(imagen)

        margenes = analizar_espacio_blanco(imagen, ancho, alto)
        if margenes[0] is None:
    
```

```

print(" No se pudo determinar el cuadro delimitador del contenido.")

margen_izquierdo_puntos = 0
margen_derecho_puntos = 0
margen_superior_puntos = 0
margen_inferior_puntos = 0

else:

    margen_izquierdo_px, margen_derecho_px, margen_superior_px, margen_inferior_px = margenes

    ancho_contenido_px = margen_derecho_px - margen_izquierdo_px
    alto_contenido_px = margen_inferior_px - margen_superior_px

    margen_izquierdo_puntos = convertir_pixeles_a_puntos(margen_izquierdo_px, dpi[0])
    margen_derecho_puntos = convertir_pixeles_a_puntos(margen_derecho_px, dpi[0])
    margen_superior_puntos = convertir_pixeles_a_puntos(margen_superior_px, dpi[1])
    margen_inferior_puntos = convertir_pixeles_a_puntos(margen_inferior_px, dpi[1])

    print(f" Cuadro del contenido: izquierdo={margen_izquierdo_px}, superior={margen_superior_px}, derecho={margen_derecho_px}, inferior={margen_inferior_px}")
    print(f" Dimensiones del contenido (pixeles): ancho={ancho_contenido_px}, alto={alto_contenido_px}")
    print(f" Márgenes (puntos): izquierdo={margen_izquierdo_puntos}, derecho={margen_derecho_puntos}, superior={margen_superior_puntos}, inferior={margen_inferior_puntos}")

    print(f" Dimensiones detectadas: ancho={ancho_puntos}, alto={alto_puntos}")

margen_ancho_puntos = min(margen_izquierdo_puntos, margen_derecho_puntos)
margen_alto_puntos = min(margen_superior_puntos, margen_inferior_puntos)

ancho_contenido = ancho_puntos - margen_ancho_puntos * 2
alto_contenido = alto_puntos - margen_alto_puntos * 2

ancho_objetivo = ancho_puntos * (1 - 2 * MARGIN_PERCENT)
alto_objetivo = alto_puntos * (1 - 2 * MARGIN_PERCENT)

escala_ancho = ancho_objetivo / ancho_contenido
escala_alto = alto_objetivo / alto_contenido

print(f" Dimensiones del contenido (puntos): ancho={ancho_contenido}, alto={alto_contenido}")

if ancho_contenido <= 0 or alto_contenido <= 0:
    print("Error: No se pudieron determinar las dimensiones del contenido.")
    return None

print(f" Dimensiones objetivo: ancho={ancho_objetivo}, alto={alto_objetivo}")

```

```

print(f" Escala de ancho calculada: {escala_ancho}, escala de alto calculada: {escala_alto}")

factor_escala = min(escala_ancho, escala_alto)
print(f" Factor de escala final: {factor_escala}")

return factor_escala

except Exception as e:
    print(f"Error al obtener las dimensiones del PDF o calcular el factor de escala: {e}")
    return None


def escalar_pdf(input_pdf, output_pdf, factor_escala):
    """Escala un PDF usando pdfjam."""
    print(f"Escalando {input_pdf} a {output_pdf} con factor de escala: {factor_escala}")
    try:
        subprocess.run(
            [
                "pdfjam",
                "--scale",
                str(factor_escala),
                input_pdf,
                "--outfile",
                output_pdf,
            ],
            check=True,
        )
        print(f"PDF escalado exitosamente: {input_pdf} a {output_pdf}")
    except subprocess.CalledProcessError as e:
        print(f"Error al escalar el PDF: {e}")
    except FileNotFoundError:
        print("Error: No se encontró el comando pdfjam. Asegúrese de que esté instalado y en el PATH del sistema")

if __name__ == "__main__":
    if len(sys.argv) != 3:
        print("Uso: python escalar-pdf.py <input_pdf> <output_pdf>")
        sys.exit(1)

    input_pdf = sys.argv[1]

```

```
output_pdf = sys.argv[2]
print(f"PDF de entrada: {input_pdf}, PDF de salida: {output_pdf}")

if not os.path.exists(input_pdf):
    print(f"Error: No se encontró el archivo PDF de entrada: {input_pdf}")
    sys.exit(1)

factor_escala = calcular_factor_escala(input_pdf)
if factor_escala is None:
    sys.exit(1)

escalar_pdf(input_pdf, output_pdf, factor_escala)
```