ARCHITECTURE & ENGINEERING DOCUMENT

DarkPDF

High-Performance PDF Dark Mode Conversion Engine

DarkPDF is a high-performance PDF dark mode conversion engine designed to transform light-themed documents into visually stable, readability-optimized dark-mode versions. Moving away from unstable semantic reconstruction methodologies, DarkPDF implements a controlled luminance-transformation philosophy. The architecture prioritizes typography preservation, rendering fidelity, lightweight execution, and broad compatibility across procedurally inconsistent PDF files, ensuring zero structural damage while maximizing long-duration reading comfort.

CORE FEATURES

Stream Mutation Engine

A lightweight rendering pipeline that mutates existing PDF color operators directly instead of rebuilding text elements or document geometry.

Grayscale Inversion System

Converts RGB and grayscale operators into precisely mapped, controlled luminance ranges calibrated for dark-background legibility.

Adaptive Tone-Curve Biasing

A targeted mathematical remapping layer engineered to dynamically enhance contrast in problematic mid-light gray regions.

XObject Compatibility

Recursively processes standard page content streams alongside embedded XObject streams to maximize structural reliability.

Structural Preservation

Preserves original glyph rendering, positioning, tracking, and complex document geometry instead of rasterizing typography.

Lightweight CLI Workflow

A highly efficient Python-based command-line interface requiring minimal external dependencies, optimized for automated pipelines.

HOW IT WORKS

The DarkPDF engine parses PDF page structures and embedded XObject streams directly using the PyMuPDF library. Instead of targeting text elements semantically, the engine operates exclusively at the graphics rendering-state level. It systematically scans for procedural rendering operators—such as RGB fill commands (rg, RG) and grayscale commands (g, G)—and remaps their associated luminance values into a controlled dark-mode grayscale color space.

The core pipeline leverages luminance-aware inversion formulas over naive bitwise or color inversion. Mid-light gray regions undergo processing through a localized tone-curve bias that amplifies luminance separation in low-contrast boundaries. Finally, a controlled dark background layer is structurally injected beneath the existing document tree, maintaining the native rendering sequence and eliminating visual artifacting.

ACTIVE STATE LOGIC

The system enforces a strict rendering-preservation-first protocol. All operations must conform to the following architectural directives:

- Preserve native rendering operators whenever possible.
- Mutate only luminance mappings; avoid geometric or semantic alterations.
- Strictly avoid semantic reconstruction, text overlays, and glyph replacement.

Tone-Curve Bias Activation Condition

To manage problematic low-contrast thresholds, a localized tone-curve adjustment activates dynamically based on the following algorithm:

```
# Dynamic Luminance Correction Loop
if 0.58 <= luminance <= 0.82:
    inverted_value *= 0.78
```

This narrow luminance-space contraction forces optimal contrast separation between the inverted foreground text and mid-light background fills, stabilizing readability without distorting global page aesthetics.

MAJOR CHALLENGES & RESOLUTIONS

1. TEXT TARGETING INSTABILITY

PROBLEM: PDF architectures routinely store typographical data as fragmented, glyph-level procedural operations rather than cohesive semantic strings. Early experimental tracking methods trying text reconstruction or structural overlays yielded broken typography and severe document corruption.

RESOLUTION: Rebuilt the pipeline completely around graphics-state mutation. By preserving the original glyph stream intact and altering only the color state parameters, structural typography errors were fully mitigated.

2. LOW-CONTRAST MID-GRAY REGIONS

PROBLEM: Standard light-themed documents often incorporate mid-light gray container shapes or background zones that, upon naive grayscale inversion, collapse into low-contrast blind spots that impair readability.

RESOLUTION: Deployed a lightweight tone-curve bias operating inside a constrained luminance evaluation window. This yields high local contrast rendering without introducing global palette distortion.

3. PDF PROCEDURAL COMPLEXITY

PROBLEM: Commercially generated PDFs display immense pipeline fragmentation, compounding subset fonts, inline XObjects, disconnected Tj operators, manual positioning rules, and highly inconsistent processing orders.

RESOLUTION: Abandoned high-level semantic stream analysis. DarkPDF explicitly decouples processing from semantic intent, maintaining focus on low-level, stable, and deterministic graphics-state transformations.

4. OVERENGINEERED DETECTION FAILURES

PROBLEM: Implementation of advanced heuristic patterns—such as spatial cluster mapping, operator adjacency tracking, and glyph correlation layers—rendered the core processing stack volatile and unpredictable.

RESOLUTION: Reverted to a streamlined, simplified V7-based architecture emphasizing low-impact mutation, luminance-only adjustments, and consistent operator mapping, resulting in absolute runtime reliability.

UI/UX & OPERATIONAL IMPROVEMENTS

- Premium Rendering Preservation: Retains perfect structural alignment, text tracking flow, and layout geometry.
- Friction Reduction: Streamlined CLI tooling minimizes integration overhead, promoting seamless batch execution.
- Visual Fatigue Minimization: Deliberately avoids stark, pure-white text mappings in favor of attenuated, softer light-gray tones.
- Persistent Contrast Optimization: The intelligent tone-bias layer continuously ensures clean contrast bands across diverse documents.

FINAL TECH STACK SPECIFICATION

Core Runtime	Python 3.x Engine
Parsing Library	PyMuPDF (fitz API wrapper)
Target Protocols	PDF Content Streams, Graphics Operators (rg, RG, g, G)
Stream Traversal	Recursive XObject Parsing & Inline Stream Traversal
Interface Layer	POSIX-Compliant Command Line Interface (CLI)