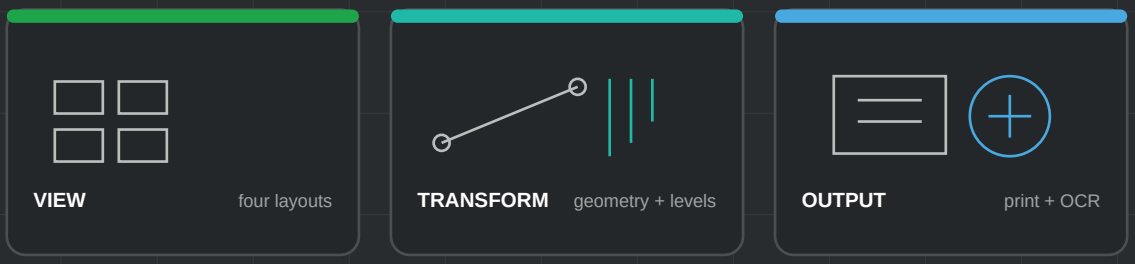
KillerPDF

Technical Specification & Complete Feature Brochure

VERSION 1.7.2



A native Windows PDF viewer, annotator and light editor.
Single self-contained executable. Built on WPF and PDFium.

This document is itself a KillerPDF showcase - open it in the app and explore the outline, forms, themes, and annotation pages.

KillerPDF.net

`.NET Framework 4.8 - x64 - Windows 10/11 - GPL-3.0`

About the Maker

KillerPDF

Steve the Killer

Field technician - systems administrator - builder of Killer Tools

KillerPDF was born out of a ticket queue.

I spend my days as a field tech and systems administrator at a large MSP, and a steady share of that work is always the same shape: a user can't fill a PDF, a form won't flatten, a license won't activate, a viewer throws an error on a file that should just *open*. Adobe Acrobat tickets - install it, repair it, license it, work around it - added up to hours every week that never felt like they were actually fixing anything.

So I built the tool I wished my users had: one fast, self-contained executable that opens the broken files, fills forms, signs documents, OCRs scans and gets out of the way. No installer, no account, no subscription. It is deliberately overbuilt for the encrypted, broken-xref and rotated-and-clipped PDFs that make other viewers give up - because those are the files that generate tickets. Everything in this brochure is real and shipping.

> "I got tired of closing the same Adobe ticket. So I wrote a PDF app that just works, ships as one .exe, and never asks the user to log in. That app is KillerPDF."

KillerPDF is free software (GPL-3.0). Find it, and everything else I build, at **KillerPDF.net**.

INTRODUCTION

About this Brochure

This document is the complete technical specification and feature list for KillerPDF version **1.7.2**, and a working showcase of the app itself.

Open it inside KillerPDF and it becomes a tour: the outline is a deep nested bookmark tree, Part IV has a real interactive AcroForm and a printed-style form to mark up, and the annotation pages are seeded for screenshots. Every tool, theme, shortcut and constant here is taken directly from the source.

WHAT CHANGED IN 1.7.2

The viewer is now fully self-contained per pane. Night-mode invert follows pane focus, Two-Page view adds a book layout, Transform adds source Levels, and Print Preview adds paper size, paper source, organized sections, live render progress, and asynchronous spooling.

The release also adds Polish localization and seven themes: 98SE, Ectoplasm, Decay, Mourning, Sepulchre, Delirium, and Malaise.

HOW THIS DOCUMENT IS ORGANIZED

- **Part I** - platform, architecture, render pipelines, view modes, and calculation details.
- **Part II** - every tool in the GUI and under the hood, including the command-line interface.
- **Part III** - keyboard map, themes, languages, robustness, standards, and hard numbers.
- **Part IV** - a live showcase with interactive forms and annotation-ready pages.

Table of Contents

Platform & Architecture

- The technology and ideas the app is built on
- Single-class WPF architecture, two render pipelines
- A zoom-stable coordinate space shared by every view

PLATFORM

Product & Tech Stack

KillerPDF is a native Windows PDF viewer, annotator, and light editor shipped as one self-contained executable. PDFium renders, PdfPig reads text, and the vendored PdfSharpCore object model writes standards-conscious output. Each document pane is a reusable PdfViewer with independent tabs, view state, render surfaces, and invert state.

Attribute	Value
Version	1.7.2 (mirrors the csproj version)
Target	.NET Framework 4.8, WPF, WinExe, x64
Distribution	Single portable exe; optional per-user or all-users install
Render / text	PDFium via Docnet.Core / PdfPig 0.1.14
Write / signing	Vendored PdfSharpCore 1.3.67 / PDFsharp 6.2.4 CMS SHA-256
Image codecs	SixLabors.ImageSharp 2.1.13
OCR	Tesseract 5.2, 11 downloadable language models
Command line	Merge, split, decrypt, rasterize, flatten, print, OCR, and batch-resave

AT A GLANCE

- Four view modes, including Two-Page book layout, share one stable annotation coordinate space.
- Split view hosts two independent document panes; night-mode invert is stored and rendered per pane.
- A full annotation, forms, OCR, signing, search, page editing, print, and Transform toolset.
- Thirteen themes, six accent variants on base themes, and eleven interface languages.

Architecture Overview

One shell, reusable document views. MainWindow owns the shared toolbar, sidebar, dialogs and application-level workflows. Each pane is a **PdfViewer** control that owns its view state, tab strip, render surfaces and document interaction code. Two control instances create split pane without a second window. Rendering, annotations, forms, links, crop, selection, text editing, tabs and zoom are divided across 16 `Controls/Viewer/PdfViewer.*` partials. Bridge files keep the remaining shell features connected while the refactor continues; cross-cutting work lives under `Services/` and feature controllers.

Two render pipelines

Single, Two-Page and Grid render into the primary tile (`PageImage` + `_annotationCanvas`) through `RenderPage(idx)`, with extra tiles built by `RenderAdditionalPages`. Continuous mode owns a separate path: `SetupContinuousView` builds one overlay per page and `RenderContinuousPages` streams bitmaps on a background thread. The continuous strip is virtualized - only a window of pages around the viewport holds real bitmaps, so a 243-page image-heavy PDF costs a few hundred MB instead of gigabytes.

```
INVARIANT RenderPage is Single / Grid / TwoPage only and no-ops in Continuous - it calls
ClearSecondaryPages, which would otherwise repoint every overlay at the hidden primary
tile and paint annotations off-screen. (The coordinate math is in the deep-dive.)
```

Per-tab sessions & the LRU cache

Each open document is a `DocumentSession` holding its zoom, fit and view mode, grid column count, tool, page, scroll offsets and the per-document dictionaries (annotations, render dims, rotations, form values, undo and redo stacks, jump history). Switching tabs re-points the live fields at the session by reference. Each session also carries a frozen-bitmap `RenderCache` keyed by (page, size-bucket, rotation), so returning to a recent tab skips PDFium entirely; whole-tab caches evict beyond the three most recent. Fit, zoom, view and page persist per file path across restarts (capped at 40 documents); lazy tabs defer loading until first activated.

View Modes

Four view modes share the same 2048-longest-side annotation space, so marks do not drift when the layout changes.

Mode	Behavior
Continuous (F5)	Vertical stream of every page; nearby bitmaps render asynchronously and distant ones leave the cache.
Single (F6)	One page in the primary tile with exact fit and zoom math for that page.
Two-Page (F7)	Facing-page spreads. Press B for book layout, which shows the cover alone and pairs the remaining pages like a physical book.
Grid (F8)	Whole-document thumbnail grid. Zoom changes the stored column count and the viewport determines the current page.

The moon button or N key inverts only the focused pane. The other pane can remain in its original colors, and focus immediately updates the rail icon. Shift+N controls whether images are included in inversion.

> F10 creates the second pane. Each pane keeps its own tabs, current page, zoom, view mode, sidebar position, selection, tool state, render cache, and night-mode invert state.

How It's Calculated: Rendering & Zoom

Two numbers describe every page: where things *are* (zoom-independent) and how many pixels we *draw* (zoom-dependent). Keeping them separate is what makes annotations stay put while text stays crisp.

1. The render-dim space

Every page is normalized so its longest side is exactly 2048 units; annotation coordinates are stored here, identical at every zoom and across all four pipelines.

```
maxDim = max(pageWidthPt, pageHeightPt)
rdW = round(2048 * pageWidthPt / maxDim)
rdH = round(2048 * pageHeightPt / maxDim) // longest side -> 2048
```

2. Decoupled bitmap resolution

The bitmap rasterizes at a resolution that follows DPI and zoom, so a 3x page is drawn from 3x the pixels rather than upscaled. The 6144 px ceiling bounds memory.

```
scaledMax = min( 6144, int( 2048 * max(dpiScaleX, dpiScaleY) * max(1.0, zoom) ) )
```

3. The coordinate Y-flip

PDF uses a bottom-left origin in points; WPF canvases a top-left origin in DIP. Every mark crosses that boundary:

```
sx = renderW / pdfW sy = renderH / pdfH
canvasX = left * sx
canvasY = renderH - top * sy // flip the vertical axis
```

rdW/rdH are rounded once and reused, so the same integers drive the primary, continuous and grid tiles - a mark in one mode lands on the same pixel in every other.

How It's Calculated: Interaction

The same precision applies to what you do with the mouse.

Cursor-anchored zoom

Ctrl+wheel zooms around the cursor by a constant 10% ratio per notch: the view scales instantly while the wheel is moving and the crisp high-resolution re-render happens once when the wheel rests. The cursor position and scroll offsets are captured *before* the zoom changes, then the offsets that keep that point fixed are applied once layout settles.

```
ratio = newZoom / oldZoom
newHOff = (oldHOff + cursorX) * ratio - cursorX
newVOff = (oldVOff + cursorY) * ratio - cursorY // clamped at >= 0
```

Grid zoom by column count

Grid snaps to a whole number of columns: the count is authoritative and the zoom is derived from it, so the grid lays out exactly N pages with no leftover gap.

```
GridZoomForN(n) = (viewportW - 24) / ( n * (rdW + 12) )
```

Cache keys & eviction

The render cache is keyed by the tuple that uniquely determines a page's pixels, so a hit is always safe to reuse; whole-tab caches drop off the end of a three-deep most-recently-used list.

```
key = ( pageIndex, sizeBucket, rotation )
sizeBucket = scaledMax (primary) or round(primaryDipW) (tiles)
evict : keep the 3 most-recently-used tabs; clear the rest
```

II

Tools & Features

- Every annotation, editing and document tool
- How each behaves in the GUI
- What happens under the hood, with real names

KillerPDF

Text & Typewriter

Text `[T / 1]`

Drop a text box anywhere, or re-edit existing PDF text in place.

IN THE GUI

Click to place a box and type (Enter saves, Escape cancels); double-click existing text to re-edit. A floating bar sets color, opacity, fill, point size, font and Bold / Italic / Strike / Underline.

UNDER THE HOOD

```
PlaceTextBox builds a flat WPF TextBox with live resize handles; CommitActiveTextBox converts it to a TextAnnotation. Re-editing drops a page-color CoverAnnotation (paired by PairId) over the original; font and size are sniffed from PdfPig with a mojibake guard, falling back to Segoe UI 14pt.
```

Highlight, Strikethrough & Underline

Highlight / Strikethrough / Underline `[H / 2]`

Three text-flowing mark styles sharing one annotation type, with an optional eraser.

IN THE GUI

Drag across words and the markup hugs each line, browser-style, instead of laying down one rectangle - across lines, paragraphs and (in continuous view) across pages. One gesture makes one grouped annotation per page that selects, moves, deletes and undoes as a unit. On pages with no text layer the tools show a status hint instead of drawing a box; the highlight eraser keeps its rectangle.

UNDER THE HOOD

```
The Select tool's drag tracks the actual character runs in reading order
(Services/TextRunService.cs, PdfPig + PDFium char boxes). CommitFlowingHighlight turns the
per-line rects into HighlightAnnotations grouped by GroupId, committed as one undo step. Each is
tagged with a HighlightStyle and RGBA color.
```

Ink (Draw) & Line

Draw & Line `[D / 5 - L / 3]`

Freehand ink and straight lines, both backed by one ink annotation type.

IN THE GUI

Draw paints freehand strokes and keeps drawing without re-arming; Line drags a straight segment with an optional Level snap. The bar offers color, opacity, width and an eraser (Draw) or level (Line).

UNDER THE HOOD

> Both create an InkAnnotation (points + stroke width + RGBA, default red 2.0px). Draw accumulates points on move; Line keeps two points. A minimum-length guard rejects click jitter.

Shapes

Shapes [4]

Rectangle, ellipse and free-form polygon markers, each with an optional fill.

IN THE GUI

Box keeps the classic drag-a-filled-rectangle gesture the highlighter used to have; ellipse and polygon are closed outlines that move, resize, flatten and print like any other drawing. Free-form places points click by click - click the first point (its target lights up as you approach) or double-click to close, Esc cancels, Backspace removes the last point. The tool shares the draw bar's color, size and opacity, with a mini-shape sub-mode picker and a Fill toggle.

UNDER THE HOOD

```
New Shapes.cs partial (EditTool.Shape + ShapeKind). Box+fill is a HighlightAnnotation; box no-fill and ellipse and polygon ride the InkAnnotation pipeline with a closed flag and optional FillR/G/B/A, rendering as a WPF Polygon and exporting via gfx.DrawPolygon (fill then stroke).
```

Image & Cover

Image [I / 6]

Place a picture on the page; covers provide whiteout for text replacement.

IN THE GUI

Pick Image, click and choose a file (or just paste with Ctrl+V); it lands at 50% scale, corner-resizable. Covers are the opaque whiteout half of an in-place text re-edit.

UNDER THE HOOD

```
ImageAnnotation stores Base64 bytes with the decoded bitmap cached, drawn at SourceWidth*Scale.
Decoding runs through SixLabors.ImageSharp 2.1.13, and images with broken DPI metadata (WhatsApp
photos, some scans) are kept within Adobe's supported 3-14,400 point page range. CoverAnnotation
derives from HighlightAnnotation with forced-opaque alpha, paired by PairId and flattened solid on
export.
```

Crop

Crop [C / 8]

Trim pages with numeric precision, on one page or all.

IN THE GUI

A default box inset 8% from the edges plus a bar with X / Y / W / H inputs, a unit selector (points, inches or %), current- or all-pages scope, Remove-crop and Apply / Cancel.

UNDER THE HOOD

> `ApplyCrop converts the canvas rect to PDF coordinates (rotation-aware via CanvasToPdfRect), clamps to the media box, writes /CropBox and /TrimBox with PdfSharpCore, then saves and reloads. Every save also strips degenerate zero-size crop boxes, healing files damaged by older builds.`

Links

Links

Existing PDF links become clickable; you can copy or remove them.

IN THE GUI

Links become hand-cursor overlays: left-click follows an internal jump or external URI; right-click offers Copy URL or Remove Link. Internal jumps are recorded in the jump history, so Alt+Left retraces them. An optional Confirm before opening links setting, on the About card beside the recent-files toggle, asks before a link opens in your browser; it is off by default so links stay immediate unless you want the check.

UNDER THE HOOD

```
RenderPageLinks reads each page's /Annots, parses /Link /Rect and the /A action and builds transparent overlays; object-stream PDFs fall back to PDFium link enumeration through a cached handle that is released before every save. On save, StripLinkAnnotationBorders removes borders so they don't strobe.
```

Forms (AcroForm)

Forms

Fill interactive text fields, checkboxes, radios and dropdowns.

IN THE GUI

Widget fields overlay the page with a cream background and accent-on-focus border; text fields show a font-size stepper as an inline flyout that follows the field through scroll and zoom. Checkboxes, radios and dropdowns are editable and written back on save.

UNDER THE HOOD

```
Widgets parse into FormFieldInfo (type /Tx, /Btn or /Ch; /Rect with full 0/90/180/270 rotation handling; /DA font size). Values persist by object number / field name and are written by WriteFormValuesToDocument, which regenerates each field's appearance and sets /NeedAppearances so other readers redraw correctly. Try the Interactive Feedback Form in Part IV.
```

Stamp, Watermark & Page Numbers

Stamp / Watermark [S / 0]

Two independent overlays - page numbers and a text or image watermark.

IN THE GUI

A live-preview editor with a page stepper. Page numbers take a start value, a format ({n} current, {N} total), size, color, a range like 1-3,5, eight positions plus custom and a spread mirror. Watermarks take text or an image, size / color or scale, opacity, angle and position. Stamps apply to a clean document with no other markup, and clear again by applying with both sections off.

UNDER THE HOOD

```
A StampSpec holds every setting (watermark default DRAFT 64pt 25% at 45 degrees).
DrawStampsOnDocument flattens via PdfSharpCore XGraphics - watermark under, numbers on top, image
opacity baked through a color matrix, rotation via translate-plus-rotate.
```

Search

Search `[Ctrl+F - F3]`

Debounced full-text search with persistent per-page highlights.

IN THE GUI

Ctrl+F toggles a draggable search bar (250ms debounce); the status shows current / total, and F3 / Shift+F3 step to the next / previous match from anywhere (F3 opens search when it isn't showing). Enter, Shift+Enter and the wheel walk the matches too.

UNDER THE HOOD

```
SearchService.Search scans each page's PdfPig GetWords() case-insensitively (single- and
multi-word) for per-page rectangles. ApplySearchHighlights repaints them on every render - the
current match a brighter accent outline, the rest dim.
```

OCR

OCR `[Ctrl+Shift+O / Ctrl+Shift+I]`

On-device text recognition in eleven languages, including Polish, with searchable-PDF output.

IN THE GUI

Ctrl+Shift+O recognizes the current page to the clipboard. Ctrl+Shift+I drags a region. The OCR menu manages installed models and can write transparent recognized text back over a scan so it becomes selectable and searchable without changing its appearance.

LANGUAGE CATALOG

English, Bengali, Czech, German, Spanish, French, Japanese, Polish, Turkish, Chinese Simplified, and Chinese Traditional. Standard and high-accuracy model tiers download on demand.

UNDER THE HOOD

> `OcrNativeBootstrap extracts the x64 Tesseract and Leptonica runtime to a per-version cache. Pages render at about 300 DPI and feed OcrService, which returns text and confidence while all recognition stays on the device.`

Digital & Visual Signing

Digital Signing (cryptographic)

Certificate-based CMS signatures with whole-chain SHA-256.

IN THE GUI

Pick a certificate from a .pfx / .p12 file or the Windows store, add an optional reason, location and contact, and write a signed copy.

UNDER THE HOOD

> Built on PDFsharp 6.2. WindowsCertificateStore enumerates the personal store; KillerCmsSigner implements IDigitalSigner on .NET SignedCms (CNG-capable, so token / cloud keys can prompt for a PIN), with SHA-256, the whole chain and a 16,384-byte /Contents reservation. Because any later edit breaks a signature's digest, resaving a signed file strips the dead signature value rather than shipping one that fails strict validation.

Visual Signatures `[G / 7]`

Hand-drawn or imported signatures, saved for reuse.

IN THE GUI

A draw pane records strokes at three pen widths; saved signatures and initials live in a draggable popup that remembers its position and drop straight into an AcroForm field.

UNDER THE HOOD

> SignatureStore persists them as JSON in %LOCALAPPDATA%. Placement creates a SignatureAnnotation (scale 0.5 full, 0.3 initials) that can bind to a form field.

Print

Print [Ctrl+P]

A live print-preview window with printer-aware paper, layout, output, and progress controls.

PRINTER

Choose the printer, paper size, and paper source. Match document automatically selects an appropriate sheet when the driver exposes one.

LAYOUT

Portrait or landscape, nine placement anchors, none-to-wide margins, 1, 2, 4, 6, or 9 pages per sheet, and fit, actual-size, or custom scaling. PRINTER, LAYOUT, and OUTPUT are collapsible sections.

OUTPUT

Color or black-and-white, copies, page ranges, odd/even subsets, and printer-supported two-sided output. Preferences for printer, orientation, color, and duplex persist.

PREVIEW AND SPOOL

Pages render progressively with a live count, navigation remains available, and Print stays disabled until the preview is ready. Final 300 DPI preparation and asynchronous spooling show progress instead of freezing the window.

> The preview and final job use the same sheet composer, so scale, margins, alignment, page subsets, N-up placement, copies, and duplex padding match what is shown.

Rotate & Transform

Rotate / Transform [R / 9]

Use Transform when a page is sideways, stretched, mirrored, crooked, photographed at an angle, or too pale to read. Every adjustment previews on the full page before anything is committed.

1 OPEN
R or toolbar slot 9

2 FIX SHAPE
rotate, skew, perspective

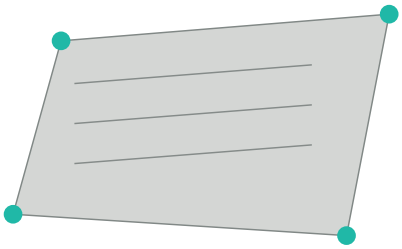
3 FIX TONE
black, white, midtones

4 APPLY
one undoable edit

WHICH CONTROL SHOULD I USE?

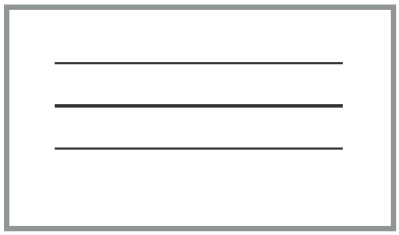
Problem	Control	What to do
Sideways or upside down	Rotate	Choose 90 degrees, or enter any angle. Expand keeps corners from being clipped.
Page is too large or small	Scale	Change width and height together, or unlock them to correct stretching.
Text reads backward	Flip	Mirror horizontally; vertical flip handles an upside-down scan without rotating.
Lines climb or fall	Skew / Level	Enter a small angle, or draw the Level line along text that should be horizontal.
Photo tapers at one side	Perspective	Place four handles on the paper corners. Page 26 walks through this in detail.
Gray paper, pale text	Levels	Move black point up, white point down, then tune midtones for readable detail.

BEFORE



crooked geometry + weak contrast

AFTER



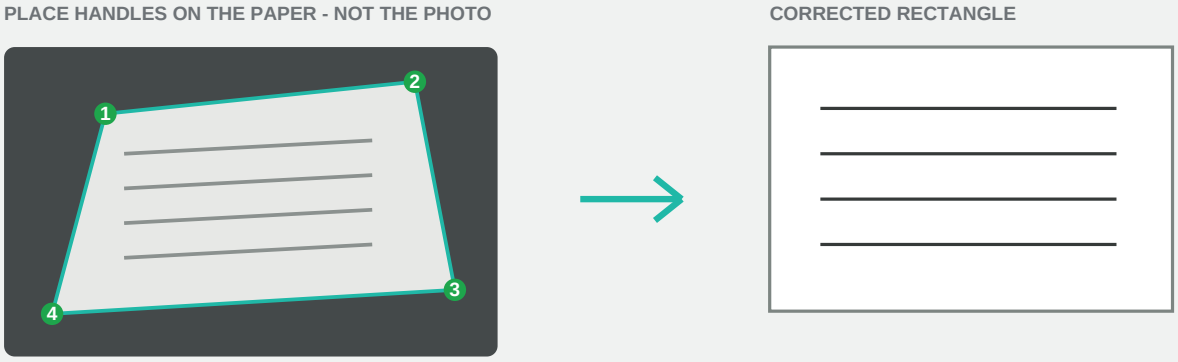
straight page + restored tonal range

> Recommended order: correct perspective first, then rotate or straighten, then scale or flip, and adjust Levels last. Reset inside one section clears only that section; Reset all returns the entire preview to the original. Apply replaces the page once, moves annotations with it, and Ctrl+Z restores the original.

Perspective Correction

Straighten a photographed page [R / 9]

Use Perspective when a rectangular page, sign, screen, or whiteboard looks narrower on one side because the camera was not directly in front of it. Do not use it for a simple tilted scan - the Level line is faster for that.



DO THIS

Step	Action
1. Turn it on	Open Transform, expand PERSPECTIVE, and enable Correct trapezoidal distortion.
2. Place corners	Drag the handles in order: top-left, top-right, bottom-right, bottom-left. Match the real paper corners.
3. Check the preview	The page should become rectangular. If it twists or folds, two handles crossed or landed on the wrong edge.
4. Apply	Apply rebuilds the page at full resolution. Ctrl+Z restores the original if the result is not right.

BEST RESULTS AND COMMON MISTAKES

If you see...	Try this
A pinched or twisted result	Keep the handles clockwise and make sure their connecting sides never cross.
A border still leaning	Zoom in and put each handle on the paper edge itself, not its shadow or the photo boundary.
A page that is only tilted	Turn Perspective back off and draw the Level line along a line of text instead.
Pale text after correction	Finish the geometry first, then open LEVELS and adjust black point, white point, and midtones.

> Under the hood: the four normalized corners form a validated, non-crossing quadrilateral. KillerPDF maps that shape to a rectangle with a projective homography, samples it at full transform resolution, and composes the correction with rotation, skew, scale, flip, and Levels as one undoable page edit.

Page Operations & Import

Page operations

Reorder, insert, delete, duplicate, extract and split pages.

IN THE GUI

Rotate, delete, insert or append blank A4 pages, move up / down, duplicate, extract or split; dragging a sidebar thumbnail reorders pages.

UNDER THE HOOD

```
All structural edits route through SaveTempAndReload(keepAnnotations:true), which rebuilds the rotation and render-dim maps and refreshes thumbnails on a background thread.
```

Export & import

Render pages to images, and open images, folders and archives as pages.

IN THE GUI

Export pages as images renders to PNG or JPEG at a chosen DPI (24-1200, default 150) with an optional page range, burning in pending annotations and stamps. Open images (JPG, PNG, BMP, GIF, TIFF) as pages; folders recurse and .zip extracts to temp, with a choice to merge into one PDF or open each in its own tab.

UNDER THE HOOD

```
Export shares the CLI's --to-image pipeline and composites over white (a --transparent flag keeps raw PNG alpha). Import uses each image's own DPI (96 fallback); a 50-file cap (MaxDropFiles) prompts and truncates.
```

Split Pane

Split Pane [F10]

Two complete document workspaces side by side in one window.

IN THE GUI

F10 opens or closes the second pane. Focus decides which pane the shared toolbar, sidebar, page list, dialogs, and commands act on. Tabs can be dragged across the divider, and each side resizes without collapsing below a readable width.

INDEPENDENT PANE STATE

Per pane	Independent behavior
Documents and tabs	Each side owns its sessions, active tab, and deferred opens.
View	Current page, zoom, mode, scroll position, and render cache.
Editing	Tool, selection, annotations, forms, links, crop, and text editing.
Night mode	N flips only the focused pane; the moon icon follows focus.

UNDER THE HOOD

> The reusable PdfViewer owns document interaction and rendering. MainWindow owns shared application chrome and routes each command through ActiveViewer, preventing state from leaking across panes.

Tabs, Outline & Thumbnails

Tabs `[Ctrl+Tab - Ctrl+W - Ctrl+Shift+W]`

Multi-document tabs with flicker-free, reconcile-in-place rendering.

IN THE GUI

Open many documents as tabs; the active tab stays visible with an overflow menu. Ctrl+Shift+W (or Close Other Tabs on a tab's right-click menu) closes every tab except the current one. Collapsing the sidebar is a smooth quarter-second slide.

UNDER THE HOOD

```
Backed by DocumentSession. The strip is reconciled in place via a session-to-element cache - no re-parenting, so no flicker on switch, drag or resize. Lazy tabs defer loading until activated.
```

Outline & Thumbnails `[Ctrl+B]`

A dockable sidebar with an editable bookmark tree and lazy thumbnails.

IN THE GUI

A dockable sidebar carries a Pages (thumbnail) tab and an Outline tab that mirrors the PDF's bookmark tree (this document has a deep one, try it here) and edits it in place: add via the row at the top, rename inline (F2), nest, reorder, retarget and delete, with Ctrl / Shift multi-select and full undo.

UNDER THE HOOD

```
LoadOutlines walks _doc.Outlines into a TreeView; titles from password-protected files are re-decoded to dodge mojibake, and bookmarks that point at named destinations resolve through the catalog name-tree. Thumbnails lazy-load on a concurrency-limited thread at ~288px.
```

Redo & Jump History

Redo / Jump history `[Ctrl+Y - Alt+Left / Alt+Right]`

Browser-style history for both your edits and your movements through the document.

IN THE GUI

Ctrl+Y (or Ctrl+Shift+Z) re-applies undone actions: annotations, text edits, stamps, clears and document-level operations alike. Alt+Left / Alt+Right, or the mouse back / forward buttons, retrace bookmark, link, jump-box and Home / End jumps exactly like a browser.

UNDER THE HOOD

```
Redo entries mirror the _undoStack records; any new edit clears the redo chain, and both stacks
live per tab in the DocumentSession. Jump history is a pair of page stacks (back / forward) fed by
RecordNavJump at every hard jump; retracing pushes the current page onto the opposite stack.
```

Zoom & Undo

Zoom `[Ctrl + - Ctrl - - Ctrl+0 - Ctrl+1/2/3]`

Cursor-anchored zoom from 5% to 500%.

IN THE GUI

Ctrl+wheel zooms around the cursor; in Grid the wheel steps the column count. An editable zoom box offers fit-width and fit-page, and Ctrl+1 / Ctrl+2 / Ctrl+3 set actual size / fit width / fit page.

UNDER THE HOOD

```
ZoomMin 0.05, ZoomMax 5.0, ZoomStep 0.15; FitMode is None / Width / Page. See the deep-dive for the cursor-anchor math.
```

Undo `[Ctrl+Z]`

One history covering annotations, pages and whole-document edits.

IN THE GUI

Ctrl+Z reverses the last change; auto-repeat is ignored so one keypress is one undo.

UNDER THE HOOD

```
_undoStack holds UndoEntry records tagged by UndoKind - single or grouped annotation, clear-page, stamp batch, bookmark edit, page snapshot or a full document byte snapshot.
```

Command Line

Every core operation also runs headless from a terminal. No window, meaningful exit codes (0 success, 1 failed, 2 bad usage), and it works even while the app is open. Each command reuses the exact pipeline its GUI equivalent runs.

```
KillerPDF.exe --merge out.pdf a.pdf b.pdf scan.jpg
KillerPDF.exe --extract-pages in.pdf 1-3,5 out.pdf
KillerPDF.exe --split in.pdf pages\
KillerPDF.exe --decrypt locked.pdf open.pdf [--password p]
KillerPDF.exe --to-image in.pdf imgs\ --dpi 300 --format jpg
KillerPDF.exe --flatten in.pdf flat.pdf
KillerPDF.exe --print in.pdf --printer "HP LaserJet" --pages 1-4
KillerPDF.exe --ocr scan.pdf searchable.pdf --lang eng
KillerPDF.exe --batch-resave inDir\ outDir\ --log report.csv
KillerPDF.exe --help
```

Batch mode & the validation harness

--batch-resave resaves a single PDF or a whole folder tree headlessly through the standard open/save pipeline with per-file OK / SKIP / FAIL reporting. It exists so the standards-conformance harness (validation/, see Part III) can push thousands of corpus files through a real KillerPDF save and prove the output validates exactly as well as the input.

Rasterizing is rotation-safe at 150 or 300 DPI, printing drives the same 300 DPI flattened pipeline as the dialog, and --version / --help print and exit cleanly for scripting.

III

Reference

The complete keyboard map and tool keys

Thirteen themes and the accent system

Eleven languages, standards conformance, and the hard numbers

Keyboard Shortcuts

Global shortcuts pause while typing in an editable text or form field. Press F1 for the in-app LIST / KEYBOARD overlay generated from the same binding map.

File	Key	Action
	Ctrl+O	Open document
	Ctrl+N	New blank document
	Ctrl+S / Ctrl+Shift+S	Save / Save As
	Ctrl+P	Print Preview
	Ctrl+W / Ctrl+Shift+W	Close tab / close others
	Ctrl+Q	Close all tabs
	Ctrl+D / F4	Document info
	Shift+F4	Show current file size

Tools	Key	Action
	V	Select
	1 / T	Text
	2 / H	Highlight
	3 / L / U	Line
	4	Shapes
	5 / D	Draw
	6 / I	Image
	7 / G	Signature
	8 / C	Crop
	9 / R	Transform
	0 / S	Stamp pages

Keyboard Shortcuts (continued)

View and navigation	Key	Action
	F5 / F6 / F7 / F8	Continuous / Single / Two-Page / Grid
	B	Book layout in Two-Page view
	F9	Cycle view modes
	F10	Split pane
	F11 / Esc	Full screen
	N	Invert focused pane
	Shift+N	Also invert images
	Home / End	First / last page
	Arrows / PgUp / PgDn	Scroll or navigate pages
	Alt+Left / Alt+Right	Back / forward jump history
	Ctrl+Scroll	Zoom around cursor
	Ctrl+= / Ctrl+- / Ctrl+0	Zoom in / out / reset
	Ctrl+1 / 2 / 3	Actual size / fit width / fit page
	Ctrl+B / Ctrl+Shift+B	Toggle sidebar / move sidebar
	Ctrl+Tab / Ctrl+Shift+Tab	Next / previous tab

Ctrl+Shift+= / - / 0 changes the whole application size. Ctrl+Shift+1 through 6 changes toolbar icon size and caption placement.

Keyboard Shortcuts (continued)

Editing, OCR, search, help	Key	Action
	Ctrl+Z	Undo
	Ctrl+Y / Ctrl+Shift+Z	Redo
	Ctrl+C / Ctrl+V	Copy / paste
	Delete	Delete selection or bookmark
	F2	Rename selected bookmark
	Enter / Escape	Confirm / cancel
	Menu / Shift+F10	Open context menu
	Ctrl+Shift+O	OCR current page
	Ctrl+Shift+I	OCR region
	Ctrl+F	Find / search
	F3 / Shift+F3	Next / previous result
	Ctrl+A	Select all
	Shift+Click	Multi-select annotations
	F1 / Ctrl+?	Shortcuts overlay
	F12	About

> Context menus show an icon for each action and print the same shortcut at the right edge whenever the action has a keyboard binding.

Tool Keys

The digits 1-9 and 0 mirror the toolbar positions left to right, and the original letters stay as aliases; tooltips append the key, e.g. `Highlight (2)`. Select is V alone - the Photoshop / Illustrator / Figma convention - which freed its digit so Shapes could take 4 and the numbers keep matching the toolbar. Esc with nothing left to cancel also drops back to Select, Acrobat-style.

Key	Action
`V`	Select
`T / 1`	Text
`H / 2`	Highlight
`L - U / 3`	Line
`4`	Shapes
`D / 5`	Draw (ink)
`I / 6`	Image
`G / 7`	Signature (opens picker)
`C / 8`	Crop
`R / 9`	Rotate (Transform window)
`S / 0`	Stamp

REFERENCE

Themes & Accents

Thirteen built-in themes share one semantic resource map. Gradient themes keep their full outer-window color in About and Keyboard Shortcuts overlays.

Dark `#1EA54C`
Charcoal and brand green

Light `#18843D`
Light surfaces and deep green

Black `#00FF66`
High contrast neon chrome

Blood `#F8C99E`
Oxblood and cream

Greed `#E0D49A`
Forest green and gold

Cyanotic `#E0D49A`
Deep blue and warm accent

98SE `#004F00`
Classic beveled Windows chrome

Ectoplasm `#EAD900`
Purple-to-charcoal gradient

Decay `#B8AA7C`
Slate-to-olive gradient

Mourning `#FF6F91`
Muted violet and pink

Sepulchre `#4FAAA8`
Charcoal-to-teal gradient

Delirium `#DD8500`
Oxblood-to-violet gradient

Malaise `#FF6F91`
Mauve-to-teal gradient

Accent variants

Dark, Light, and Black also accept Green, Red, Orange, Teal, Blue, and Purple overlays.

Localization

KillerPDF ships eleven interface languages. English remains the base dictionary, so a missing translated key falls back safely while theme and language switches repaint live.

Locale	Name
English (en-US)	English
Spanish (es)	Español
German (de-DE)	Deutsch
French (fr-FR)	Francais
Turkish (tr-TR)	Turkce
Bengali (bn)	Bangla
Japanese (ja-JP)	Nihongo
Chinese Simplified (zh-CN)	Simplified
Chinese Traditional (zh-TW)	Traditional
Czech (cs-CZ)	Cestina
Polish (pl-PL)	Polski - new in 1.7.2

LocaleManager merges theme + en-US + the chosen locale. Code-built context menus, toolbar captions, annotate bars, dialogs, and the shortcuts overlay rebuild when the language changes. OCR exposes the same eleven-language catalog, including Polish.

File Handling & Robustness

KillerPDF is built to open the PDFs other viewers choke on, and to never lose your work to a malformed file.

Resilient open

- Network / UNC paths are copied locally first to avoid 9P short-read corruption; owner-restricted /Encrypt PDFs prompt for a password.
- An exception-driven fallback chain: xref corruption falls back to read-only then a PDFium repair; FlateDecode / EOF errors strip encryption and re-save losslessly; a zero-page page tree is guarded instead of crashing; anything else offers a repair.

Rotation-safe temp reload

SaveTempAndReload zeroes each page's /Rotate before saving (PDFium sizes from the unrotated MediaBox, so rotated content would clip) and restores it in memory; on xref failure it falls back to a PDFium save with FPDF_REMOVE_SECURITY or a PdfSharpCore import-repair.

Clean save & crash reporting

Saving writes a clean, annotation-free snapshot, burns stamps then annotations into a copy via PdfSharpCore XGraphics, and reopens the clean copy so future saves never double-burn. Every save also scrubs the hazards that corrupted files in older builds: dangling /Outlines references, degenerate crop boxes and dead signature values, and the cached PDFium link handle is released first. A 50-entry timestamped status ring buffer is dumped with structured crash logs to %LOCALAPPDATA%, auto-rotated at a 20 MB cap.

Standards Conformance

Every release is validated against veraPDF, the industry reference validator, across a 2,907-file public conformance corpus. The bar is zero regressions.

Every corpus file (the veraPDF test corpus for PDF/A-1/2/4 and PDF/UA-1/2, the Isartor PDF/A-1b suite and the TWG files) is validated pristine, resaved through the normal save pipeline via --batch-resave, and validated again. Any file that fails a rule after the resave that it did not fail before counts as a regression. A second pass runs qpdf --check on every original / resave pair.

Outcome (veraPDF 1.30.2)	Files
Corpus total	2,907
Resaved and revalidated	2,236
Refused (encrypted or unparseable; source untouched)	671
Conformance regressions	0 (one documented PDF 2.0 header limitation)
Files that came out more conformant	63
qpdf structural check worsened	0 (195 improved)

Patched at the source

Getting to zero meant fixing the write engine itself. PdfSharpCore is vendored under third_party/ with six patches, each marked in the source with the ISO clause it satisfies: no Producer/Creator stamping into an imported document's Info dictionary, no /ModDate rewrite at open (both break PDF/A's Info-to-XMP equivalence rule, ISO 19005-1 6.7.3), no transparency /Group injected onto every page (forbidden in PDF/A-1), stream /Length always matching the exact byte count, booleans written as the lowercase true / false keywords, and the debug verbose file layout removed.

The full report, the comparison script and reproduction instructions ship in the repo under validation/RESULTS.md; the summary lives on the technical page at KillerPDF.net.

Notable Engineering Details

A few engineering decisions worth calling out (the math behind the first two is in the deep-dive):

- **Zoom-stable coordinates.** Every annotation lives in the 2048-longest-side render-dim space, identical across all view modes, so marks never drift.
- **Decoupled bitmap resolution.** Pixels scale up to min(6144, 2048 * dpi * zoom) for sharpness while coordinates stay zoom-independent.
- **Continuous-view virtualization.** Only a window of pages around the viewport holds real bitmaps; released slots keep their exact height so scroll geometry never changes.
- **Per-tab LRU bitmap cache.** Frozen bitmaps keyed by (page, size-bucket, rotation), evicting whole tabs beyond the three most recent.
- **Reconcile-in-place tab strip.** A session-to-element cache diffs the strip instead of rebuilding it - no flicker on switch, drag or resize.
- **A vendored, patched write engine.** PdfSharpCore lives in the repo under third_party/ with six standards-conformance patches, proven by a 2,907-file veraPDF harness that runs on every release.
- **Two PdfSharp engines.** PdfSharpCore for the app, PDFsharp 6.2 (separate namespace) only for signing, behind a portable certificate-provider abstraction.
- **Self-extracting single exe.** Costura embeds managed DLLs; native Tesseract / Leptonica self-extract to a per-version cache and language packs download on demand.

REFERENCE

Specifications & Constants

Selected limits and defaults drawn from the 1.7.2 source.

Specification	Value
Render coordinate space	2048 DIP longest side
Bitmap cap	6144 px
Print / OCR render	300 DPI on demand / about 300 DPI
Image export	24-1200 DPI, PNG or JPEG
Zoom range	5% to 500%
App size	70% to 250%
Page bitmap cache	About 160 MB per tab, budgeted in bytes
Per-file state	40 documents
Folder / zip import	50 files per drop
Signature contents	16,384 bytes, CMS SHA-256
Default text	Segoe UI, 14 pt
Crash logs	50 status entries; 20 MB rotation
UI languages / themes	11 locales / 13 themes plus accent variants

The values above are compiled into KillerPDF 1.7.2 or derived directly from its current resource catalogs.

Glossary

Plain-language definitions for the terms used throughout this brochure.

Term	What it means
WPF	Windows Presentation Foundation, Microsoft's native desktop UI framework
partial class	A C# feature that lets one class be written across several files
PDFium	Google's open-source PDF rendering engine that turns a page into pixels
Docnet.Core	The .NET wrapper KillerPDF uses to drive PDFium
PdfPig	A .NET library that extracts text and word positions from a PDF
PdfSharpCore / PDFsharp	Libraries that write PDFs: saves and edits (vendored and patched), and signing
vendored	A library's source copied into the repo and built with the app, so it can be patched and audited
veraPDF	The industry reference validator for the PDF/A and PDF/UA standards
PDF/A	The ISO standard for archival PDFs; strict rules so files stay readable for decades
Tesseract	The open-source OCR engine that recognizes text in images
DIP	Device-independent pixel, a unit that stays the same apparent size on any display
render-dim	The internal page size (longest side 2048) where annotations are stored
xref	The cross-reference table, a PDF's index of where each object lives
MediaBox	The PDF entry that defines a page's full size
AcroForm	The PDF standard for interactive, fillable form fields
CMS / PKCS#7	The standard cryptographic container that holds a digital signature
OCR / DPI	Optical character recognition / dots per inch (300 DPI is print quality)

IV

Live Showcase

- A real interactive form to fill and save
- A printed-style form to mark up by hand
- Annotation pages seeded for screenshots

Interactive Feedback Form

Real AcroForm fields - open in KillerPDF (or any reader), fill them in, then Save.

Name

Role

IT / MSP technician

Email

Version used

1.7.1

Which tools do you use most? (tick all that apply)

- ☐ Annotation
- ☐ Forms
- ☐ OCR
- ☐ Signing
- ☐ Shapes
- ☐ Crop / pages
- ☐ Search
- ☐ Print

Overall rating

- ○ 1
- ○ 2
- ○ 3
- ○ 4
- ◉ 5 (best)
- ☐ Notify me of releases

Comments

Signature

Use the Signature tool (G) in KillerPDF to sign here.

Date

Sample Application Form

A printed-style form with no interactive fields - ideal for demoing fill-by-annotation, crop and stamp.

KILLERPDF - USER REGISTRATION & EVALUATION

Form KP-1 (Rev. 1.7.2) - Please print clearly in black ink

LAST NAME _____ FIRST NAME _____

ORGANIZATION _____ EMPLOYEE ID _____

EMAIL ADDRESS _____

PHONE _____ DATE (YYYY-MM-DD) _____ LICENSE No. _____

Edition requested: ☐ Portable (.exe) ☐ Installer ☐ Source build (GPL-3) ☐ Enterprise / MSP

Platform: ☐ Windows 10 ☐ Windows 11 ☐ Server 2019+ ☐ Other _____

Comments

SIGNATURE OF APPLICANT _____ DATE _____

LIVE SHOWCASE

Annotation Playground

Seeded content for screenshots. Reach for the toolbar (or the number keys) and mark this page up.

Highlight me (H / 2)

Drag the Highlight tool across this sentence and the mark flows along the words; Strikethrough and Underline give the other two styles.

Draw a shape (4)

Reach for the Shapes tool and drop a rectangle, ellipse or free-form polygon over this row, with an optional fill.

Draw here (D / 5)

Freehand ink area - round caps, adjustable width, optional eraser. The minimum-length guard ignores accidental taps.

Type a note (T / 1)

Click with the Text tool to drop a typewriter note, then style it from the floating bar.

Mark up a table

Tool	Shortcut	Output type
Highlight	H / 2	Highlight annotation
Shapes	4	Ink / highlight (filled)
Draw	D / 5	Ink annotation
Stamp	S / 0	Flattened overlay
Crop	C / 8	CropBox / TrimBox

LIVE SHOWCASE

Annotation Playground II

Seeded content for screenshots. Reach for the toolbar (or the number keys) and mark this page up.

Highlight me (H / 2)

Drag the Highlight tool across this sentence and the mark flows along the words; Strikethrough and Underline give the other two styles.

Draw a shape (4)

Reach for the Shapes tool and drop a rectangle, ellipse or free-form polygon over this row, with an optional fill.

Draw here (D / 5)

Freehand ink area - round caps, adjustable width, optional eraser. The minimum-length guard ignores accidental taps.

Type a note (T / 1)

Click with the Text tool to drop a typewriter note, then style it from the floating bar.

Mark up a table

Tool	Shortcut	Output type
Highlight	H / 2	Highlight annotation
Shapes	4	Ink / highlight (filled)
Draw	D / 5	Ink annotation
Stamp	S / 0	Flattened overlay
Crop	C / 8	CropBox / TrimBox

KillerPDF

Colophon

DESIGN & DEVELOPMENT

KillerPDF is designed and built by Steve the Killer.

ICON

Designed and drawn by Steve the Killer in GIMP.

WORDMARK

Typewriter A602 with the hand-finished KillerPDF treatment.

BROCHURE

Updated for v1.7.2; features and constants are drawn from the application source.

LICENSE

KillerPDF is free software released under GPL-3.0.

Body type is Arial. Technical callouts use Courier.

KillerPDF.net