Kuonen.PDFTools — Roadmap

Farb-Legende (Status):

✓	◑	○	★	✗
fertig	teilweise	offen	neu / Baustelle	entfaellt / nie

Diese Datei fuehrt nur noch das OFFENE. Alles Erledigte steht nach Version in eigenen

Dateien, damit die Restliste lesbar bleibt:

v4 erledigt · v3 ·

v2 · v1.

Die Spalte offen zaehlt die Zeilen mit oder in dieser Datei -

sie ist damit nachrechenbar und kann nicht mehr veralten.

Stand: aktueller Code M1565. Git-versionierte Kontroll-Roadmap; das Download-PDF (/download/roadmap) wird ab M1188 direkt aus dieser Datei erzeugt. Alles steht als Tabelle, kaum Fliesstext.

Aufwand: S klein · M mittel · L gross · XL sehr gross. Status: Fertig · Teilweise · Offen · Zurueckgestellt · Neu/Baustelle · x Nie im PDF. Aufbau: v4 = geplante Werkzeug-Paritaet (Acrobat/InDesign/Word/Excel/PowerPoint/FOP/LaTeX/HTML/Dreamweaver/Notepad + Bildformate/Schriften/Editor) + externe Punkte · v3 = abgeschlossene grosse Tracks · v2 = History · v1 = Archiv je Bereich. Verbindliche Wahrheit je Meilenstein: das Ledger in CLAUDE.md.

v4 · Werkzeug-Paritaet - geplant (offen)

Ehrliche Lueckenanalyse im Code geprueft (nicht aus dem Gedaechtnis): was Editor/Bibliothek gegenueber den grossen Vorbildern noch WENIGER koennen. Alles baubar (Grundprinzip: keine „bauen wir nie"-Grenzen). Bereits Gebautes ist bewusst NICHT gelistet (z. B. verschachtelte/GREP-Formate, Mehrspaltensatz, Silbentrennung, Index/TOC/Querverweise/Fussnoten, Data Merge, IDML — schon da). Gruppen: AP Acrobat · LXL Excel · LFO FOP · LTX LaTeX · LHT HTML · LPP PowerPoint · IMG Bildformate · FNT Schriften · TXT Notepad/Klartext · AG Apogee-Rest · ED Editor-Paritaet · PRN virtueller Drucker · LR Lightroom · LVS Visio · ARC Archive · KDX .kdoc-Vollabbild · KMU Musik im .kdoc · BSP Beispiele und Erreichbarkeit.

v4 auf einen Blick

Zwei verschiedene Zahlen — nicht verwechseln. „Punkte" zaehlt nur die Restliste unten (v4 listet ausschliesslich Luecken; alles Gebaute steht in v1–v3). „Abdeckung" ist die grobe Einschaetzung, wie weit das Werkzeug insgesamt ersetzt ist — eine Gruppe kann 0 erledigte Punkte haben und trotzdem 85 % abdecken.

Grp	Vorbild	n	offen	Abdeckung (grob)	Groesste Luecke
AP	Acrobat Pro	7	0	~97 %	alle Punkte angegangen; Reste sind in den Zeilen benannt
LID	InDesign	6	0	100 %	Fertig (M1430). Zuletzt die hilfslinienbasierte Liquid-Layout-Regel: Hilfslinien sind jetzt Teil des Seitenmodells (je Seite, im .kdoc), fluessige Linien sind Naehte, die die Groessenaenderung aufnehmen
LWD	Word	6	0	100 %	Fertig (M1431, Grenzen geschlossen M1432). Zuletzt Formen, Textrahmen, Gruppen, SmartArt und eingebettete Diagramme — gezeichnet statt uebergangen, ueber das neue gemeinsame Modul DrawingMl (auch von PowerPoint und Excel benutzt)
LXL	Excel	6	0	~92 %	Nichts mehr offen — Diagramme und Bilder werden seit KDX23/KDX27 auch geschrieben (samt Pivot, Kopf-/Fusszeilen und Blattschutz); die Grenzen stehen in den archivierten Zeilen
LFO	Apache FOP	5	0	100 %	Fertig. LFO1–LFO5 komplett (goldener Standard)
LTX	LaTeX	6	0	100 %	Fertig (M1428). Die zwei letzten Zeilen sind geschlossen: pgfplots traegt eigene Achsengrenzen und Teilstrichlisten, parametrische Plots, Balken- und Fehlerbalken-Diagramme mit Markern, die Baeume grow= in Grad samt eigenen Abstaenden und den rechtwinkligen Kantenverlauf, und ein eigener edge from parent path wird ueber den vorhandenen Pfadleser ausgewertet statt gemeldet. Dazu der .bst-Interpreter — die vier Originalstile aus TeX Live laufen durch und treffen ein echtes BibTeX 0.99d
LHT	HTML/CSS	6	0	100 %	Fertig (M1468). JavaScript vollstaendig (M1467), echte 3D-Perspektive samt preserve-3d-Szene und Tiefenordnung, negative Raender und Randzusammenfall nach CSS 2.1 §8.3.1 (M1468) — je gegen Edge geprueft

Grp	Vorbild	n	offen	Abdeckung (grob)	Groesste Luecke
LPP	PowerPoint	11	1	~99 %	M1420 hat die ganze Schreib-Seite gebaut (Layouts/Platzhalter, alle prstGeom-Formen mit Verlauf+Schatten, Aufzaehlungen mit wirksamen Tabulatoren, Diagramme), M1474 die Diagrammart aller neun Arten ueber den Rundlauf gerettet und dabei das fehlende Pflicht-Achsenpaar nachgetragen, ohne das PowerPoint 7 von 9 Dateien gar nicht oeffnete. Offen (LPP10): die Ausschmueckung der Diagramme (Achsentitel, Legende, Beschriftungen, Reihenformat) und SmartArt ohne mitgelieferte Zeichnung
LVS	Visio	44	1	~98 %	LVS25 (M1501): Datenquellen-Zeichnungen ohne Nachbearbeitung — ERD-Enden, Kanten-Anschluss, Titeltext-Format, freie Texte/Rahmen, Baum ohne Umbruch, Seite passend zum Inhalt, Linien folgen beim Verschieben, vsdx_describe mit Zellwerten/Klebungen, vsdx_check (g)-(i). Offen: LVS25.20 (Linienende am Objekt entlang ziehen, wartet auf Visio-Rueckmeldung). Bis M1437 fertig. M1429 Kern, M1434 Geometrie-Vererbung/Stil/Text/Gruppen/Bildexport, M1435 Kurven/Gruppen/Ebenen/Klebepunkte/.vdx/.vsd 11, M1436 Bilder/Fuellungen/Themen/Ebenen/.vsd 5-6/Massstab, M1437 zuletzt Bilder in binaeren .vsd, Verlaeufe in jeder Richtung samt Stopp-Transparenz und Mitdrehen, weicher Schatten, Themen-Effekte (PDF, .kdoc, Rueckexport) und Stilvorlagen im Zeichnungsmassstab. Was bleibt, sind benannte Grenzen (je zur Laufzeit gemeldet). M1465: vsdx_check/vsdx_describe/vsdx_expect, model_to_vsdx, kdoc_schema, Text auf den Textblock beschnitten; M1466: model_to_vsdx zeichnet ein ganzes Datenquellen-Blatt in einem Aufruf (Kopfkasten, Farben je Teil, Gruppen, Linien hinter den Kaesten an die Gruppe geklebt, A4-Seiten, Baum-Layout, farbige Textteile, Uebergangsmarken, Selbsttest)

Grp	Vorbild	n	offen	Abdeckung (grob)	Groesste Luecke
ARC	Archive (ZIP/RAR/7z/TAR/ ISO/DMG)	41	8	~97 %	M1423–M1433 haben alle Formate lesbar gemacht, M1488/M1489 die mehrteiligen Archive, den UDF-, LZX- und BZIP2-Packer, El Torito und die verschluesselten RAR-Koepfe. Offen: die Packer fuer Apple Archive, StuffIt und .dmg (ARC22/27/28), verschluesseltes APFS (ARC29), die RAR4-Schluesselableitung (ARC33b, ohne Orakel), der stromweise Weg fuer sehr grosse Archive samt ZIP-Verbund ueber 4 GB (ARC30/36b) und die rechtliche Einschaetzung der StuffIt-Konstanten (ARC31). RAR schreiben bleibt aus Lizenzgruenden ausgeschlossen
IMG	Bildformate	34 (davon 6 als Gruppen mit 50 Unterpunkten)	5	Lesen ~99 % · Schreiben ~98 %	Lesen ist fast vollstaendig: AVIF bit-genau gegen dav1d (IMG6 8/8), JPEG 2000 (IMG32) und JBIG2 (IMG23 8/8) sind geschlossen. HEIF/AVIF schreiben ist fertig (IMG7 11/11, M1496): eigener AV1-Intra-Codierer auf dem Syntaxweg des Dekodierers, verlustfrei und verlustbehaftet, 8/10/12 Bit, bit-genau gegen dav1d; HEIC und AVIF mit 10 Bit ueber das neue Bildmodell Rgba64Image. Offen: JPEG XL VarDCT und Schreiben (IMG8 4/6), das verlustfreie arithmetische JPEG SOF11 (IMG29). Ehrlich benannt, nachgemessen in M1496: .avif bleibt aus DocumentConverter draussen — libavif 1.4.2 (Pillow) schreibt jede gemessene Stufe (Qualitaet 50/75/90) mit Quantisierungsmatrizen, die der Leser nicht hinterlegt hat; die eigenen AVIF-Dateien liest er, fremde verlustbehaftete nicht
FNT	Schriften	7	0	Satz ~98 % · Datei-Ausgabe ~90 %	komplett — FNT4 (voller Befehlssatz + Rasterizer-Anbindung, opt-in) und FNT7 (bereits M1316 gebaut, hier nur nachgezogen)
FED	Schrift-Editor	20	0	Lesen ~97 % · Schreiben ~100 %	20/20 fertig, Format-Matrix komplett. FED18 (TTX) geschlossen — zusammengesetzte Glyphen, die CFF -Umriss-Tabelle und die post-Namen; dazu die zwei letzten Matrix-Luecken: Bitmap-Emoji schreiben (ColorBitmapFontEncoder: CBDT/CBLC + sbix) und variables CFF2 schreiben (Cff2Writer, das CFF-Gegenstueck zu FED9)

Grp	Vorbild	n	offen	Abdeckung (grob)	Groesste Luecke
LIC	Lizenz-Manager	7	0	100 %	Fertig (M1427). Die Lizenz benennt jetzt Gruppen und einzelne Werkzeuge, eine Stelle entscheidet, die Aussteller schneiden mit derselben Auswahl zu und lehnen einen Tippfehler ab statt eine tote Lizenz auszustellen, und der Pruefstand folgt derselben Lizenz. Der Wortschatz liegt als Licensing/LicenseCatalog im Kern — nicht im MCP-Server, den die Aussteller gar nicht referenzieren
FS	Datei- und Ordner-Werkzeuge	8	0	100 %	Fertig. M1424 hat die Gruppe gebaut (sieben Werkzeuge, der geteilte Lauf, die Doppel-Schranke vor den aendernden Punkten); M1425 hat den FS6-Rest geschlossen — Store-Paket-Verweis, Ressourcen-ID und die ausgelagerten .mun-Ressourcen. Es bleiben nur noch Grenzen, die Windows selbst setzt (Signatur-Pruefung, SACL, Unix-Besitzer nach Namen)
FCR	Schrift-Erzeuger	22	0	100 %	Gruppe fertig (M1512). FCR14 erledigt (M1512): Farbschriften bearbeiten (Paletten, COLR v0/v1, SVG, Bitmap-Stufen). FCR13 erledigt (M1511): variable Schriften bearbeiten (Achsen, Schnitte, Instanzen, Zwischenschnitt am Regler), Eck-Schnitte fuer gvar und CFF2, Zerlegen bestehender variabler Schriften. FCR18 erledigt (M1510): die Probe formt (GSUB/GPOS), bricht um und zeigt das Bildschirmbild mit Hinting. FCR22 erledigt (M1509): angedockte Komponenten werden gelesen (gespiegelt nach Kuonen.UI). FCR19 erledigt (M1508): Schriften ueber REST, eine gemeinsame Bearbeitungs-Anfrage fuer MCP und REST, zusammengesetzte Glyphen nachgefuehrt, name je Sprache mit Mac OS Roman, OS/2-Kennung und Hoehenpruefung, senkrechte Metriken, rueckwirkende Kerning-Klassen; neu FCR22 (Punkt-Andocken beim Lesen, gespiegelte Leser). FCR21 erledigt (M1507): Zeichnungen kommen vollstaendig an (SVG-Grundformen, Text, transform, evenodd; PDF jede Seite, Linien als Flaeche), ein neuer Platz bekommt gleich Name und Zeichen, auch in CFF. Neu, auf Nutzerwunsch: ein echter Font-Creator/-Editor in der Darstellung des grossen Editors. Die Rechenkerne liegen aus FED vor (Umrisse lesen/aendern/schreiben, Schrift von Grund auf bauen, Metriken, Kerning, variable Achsen, Farbschriften) — was fehlt, ist das Werkzeug: heute gibt es nur eine Formularseite, auf der sich EIN Umrisspunkt ziehen laesst

Grp	Vorbild	n	offen	Abdeckung (grob)	Groesste Luecke
MUS	Notensatz	17 (davon MUS13 als Gruppe mit 25 Unterpunkten)	14	Fundament ~85 %	Modell/Stich/PDF/MIDI stehen, Notentext/Gesangstext (MUS5/MUS6), die gebuendelte Notenschrift Bravura (MUS2), die feinen Stichregeln (MUS3), mehrere Notenzeilen je Stimme und der Stimmenauszug (MUS7). Der Editor bedient seit KMU19/KMU20 Akkorde, zwei Stimmen, beide Klavierzeilen, Dynamik, Artikulation, Boegen, Gesangstext und Tonartwechsel (MUS13 16/25). Die Notenblatt-Erkennung (MUS16) ist seit M1494 fertig: am echten Scan Auftakt und 16 Takte in zwei Stimmen, die Melodie Note fuer Note richtig
TXT	Notepad	6	0	~98 %	komplett; C#-Hervorhebung ohne Format-Angabe im Interpolationsausdruck
AG	Apogee	4	0	~98 %	komplett

Grp	Vorbild	n	offen	Abdeckung (grob)	Groesste Luecke
KDX	.kdoc als Vollabbild	29	0	100 %	Auf Nutzerwunsch: Import -> .kdoc -> Export ins selbe Format soll nichts verlieren. M1438-M1441 haben die nie gespeicherten Laufzeitfelder geschlossen, den PDF-Import vollstaendig gemacht und die Fliesstext-, SVG-, Metadatei- und ODG-Wege vom PDF-Umweg geloest; M1446-M1450 haben den Fliesstext-Export (docx/odt/epub/rtf/fb2/txt, neu html und md) und den Visio-Rest geschlossen; M1455 hat den Metadatei-Rest geschlossen (MetafileFrame, byte-genauer EMF/WMF-Rundlauf); KDX13 (Alt-/ActualText, Rendermodus, Clip-Pfad, Kachelmuster/Mesh-Schattierung, OCG-Verschachtelung) ist mit dieser Etappe geschlossen; KDX12 (Spiegelung an allen Rahmen) ist geschlossen (der gemeinsame Zeichen-Rahmen LayoutObject.ApplyObjectTransform, PageCanvas.MirrorAround, FlipH/FlipV an allen 16 Aufrufstellen, im .kdoc gesichert); KDX11 (Originalteile-Speicher: docProps/custom.xml aus einem .docx wird unveraendert mitgefuehrt und beim Export wieder eingesetzt, samt Rundlauf-Waechter fuer .docx) ist geschlossen - bewusst nur diese eine, sicher mitfuehrbare Teilart, siehe Zeile unten im Archiv; offen sind PowerPoint, Excel, InDesign und die restlichen Modell-Punkte. M1462 hat vier Restposten geschlossen: die mehrstufige Liste im Fliesstext samt der .kdoc-Luecke darunter (KDX15), die Sprache je Lauf beim PDF-Import (KDX16), den Export-Andockpunkt und die Hygiene der Fundgrube (KDX17) und die Stellen in der Export-Sicht (KDX19, noch nicht ganz). M1469 hat KDX5 (PowerPoint direkt) geschlossen - bis auf den kleinen Rest KDX20. M1470 hat KDX6a (Excel direkt) geschlossen: der Tabellen-Arbeitsstand liegt als eigener .kdoc-Datenteil xlsxWorkbook (Blatt, Zellwert mit Typ, Formel, Zahlenformat, Zellformat, Verbund, bedingte Formatierung, Datenpruefung, Namen, Fixierung/Filter), die Seiten sind die Ansicht - gegen Excel 16 UND LibreOffice abgenommen; offen bleibt KDX6b (Pivot, Zellanker, Diagramm-Quellbezug, Zellformat beim Schreiben). KDX7a (InDesign nativ) ist geschlossen: .idml geht in beide Richtungen direkt, mit dem neuen IdmlNativeReader, dem Bestand idmlResources (.kdoc-Schema §8.29) und genau EINEM IDML-Leser und -Schreiber; gegen das Fremdorakel SimpleIDML und drei echte InDesign-Pakete abgenommen. Offen bleibt KDX7b

Grp	Vorbild	n	offen	Abdeckung (grob)	Groesste Luecke
					(Musterseiten anwenden und zurueckschreiben, Textrahmen-Verkettung, Anker an seiner Zeile, Tabellen-Feinheiten, Kurven). Damit ist kein XL-Punkt mehr offen. Spezifikation docs/superpowers/specs/2026-09-19-kdoc-vollabbild.md M1492 hat KDX26 geschlossen (Schlagschatten, XML-Tagging, Anker an der Zeile, Tabelle neben Text) - die Gruppe ist fertig.
KMU	Musik im .kdoc	33	10	~70 %	23 von 33 erledigt (bis M1489): die Partitur liegt im .kdoc samt Stech-Regeln und Notenschrift, Wiederholungen/Volten, Spielanweisungen an beliebiger Zeitstelle, Tabulatur, N-tolen mit Pausen, Wiedergabe als MIDI, Instrumentenwechsel und ein Editor mit 44 Feldern je Eintrag. Offen: historische Notation (KMU13, XL) und neun Restposten-Zeilen, die je zur Laufzeit gemeldet werden
ED	Editor (alle Vorbilder)	25	0	InDesign ~80 % · Dreamweaver ~85 % · Acrobat, Word, Excel und PowerPoint je mit eigenem Modus (ED37–ED49), nicht neu geschaetzt	Fertig (M1502). Der Word-Fliesstext-Modus ist vollstaendig: Schreibstrich im Satzbild, Fussnoten, Inhaltsverzeichnis und Formeln (ED43a), dazu Zwischenablage, Schreibstrich in Tabellen/Bildunterschriften/Formeln, Fliesstext zwischen festen Seiten mit echten Seitenzahlen und Notentexten am Seitenfuss, und der uebernommene Fliesstext als anfassbarer Rahmen auf der Leinwand (ED43b). Die frueheren Schaetzungen (Excel ~5 %, PowerPoint 0 %, 'die Oberflaeche fehlt') stammten aus der Zeit vor Tabellenblatt-, Folien- und Praesentier-Modus (ED47–ED49) und sind entfernt

Grp	Vorbild	n	offen	Abdeckung (grob)	Groesste Luecke
LR	Lightroom	24	7	~84 %	Entwicklung, Vorgaben, Export, XMP-Sidecar, Stapel, Rauschreduzierung, Objektivkorrektur, Katalog, Galerie und Fotobuch stehen (bis M1488). Seit M1491 lokale Anpassungen mit Masken (LR7a), seit M1495 die Bereichsreparatur (Kopieren/Heilen per Gradientenangleich, Quelle automatisch), der Pinsel im Sidecar, Mittelpunkt/Rundheit, Weiss/Schwarz/Dunst/Tonung lokal und die Synchronisierung lokaler Anpassungen (LR7b). Seit M1499 Lightrooms Masken-Gruppen (LR7c: crs:MaskGroupBasedCorrections lesen und schreiben, Abziehen/Schneiden/Umkehren nach Camera-Raw-Messung, Pinsel-Gruppen, beidseitiger Verlauf, Luminanz-/Farb-/Tiefen-Bereichsmasken, KI-Masken mit gesetztem Raster). Seit M1500 Lightrooms uebrige lokale Regler (LR7e: Struktur, Farbton, Koernung, Rauschen, Moire, Farbsaum, lokale Kurve mit Saettigung verfeinern, Punktfarbe, Farbkorrektur — mit neuen globalen Bausteinen). Offen: Lightroms .acr-Raster (LR7d, nicht belegbar ohne freie Beispieldatei), LocalGlow (LR7f, Wirkung nicht belegt), GPU (LR14), RAW-Feinheiten (LR1) und drei Restposten-Zeilen
PRN	virtueller Drucker	8 (davon PRN3 als Gruppe mit 7 Unterpunkten)	2	~95 %	alle acht Punkte gebaut, und die benannten Grenzen der ersten Runde geschlossen (M1391): PCL mit Delta-Row-Kompression, Symbolsaetzen, Raendern und eingebettetem HP-GL/2; PWG mit CMYK, 16 Bit und CIELab; der Auftrag traegt seine echte Herkunft; Cancel-Job greift ueber den zweiphasigen Create-Job/Send-Document-Weg; das Kontingent uebersteht einen Neustart; Anmeldung und ipps://. Es bleiben nur noch Grenzen, die die Maschine setzt (kein Windows-Drucksystem, kein GhostPCL zum Gegenlesen) Seit M1409 auch PRN3 vollstaendig (7/7): mDNS mit Sondieren/Konflikt/Abmelden, WSD als Windows-Weg, IPP Everywhere, und der gemessene Netzweg. Was der Code kann, ist damit gebaut; was fehlt, ist der Nachweis am fremden Geraet — abgenommen ist gegen echtes CUPS (treiberlose Einrichtung + zwei angekommene Auftraege), ungeprueft bleiben Windows, macOS, iOS und Android (kein zweites Geraet im Netz). Dazu zwei benannte Luecken: WS-Print (die WSD-Druckseite) ist nicht gebaut, und auf einem Windows-/macOS-Wirt traegt keine Docker-Betriebsart die Bekanntmachung ins LAN — dort laeuft sie ueber den wirtsseitigen Ankuendiger.

Grp	Vorbild	n	offen	Abdeckung (grob)	Groesste Luecke
ROB	Erschoepfung durch begrenzte Eingabe	6	0	100 %	Fertig (M1506). ROB1–ROB4 geschlossen (M1488), der Waechter mit Zeitbudget (ROB5) steht seit M1453, und seit M1506 laufen auch die Schrift-Leser hindurch (ROB6) — samt einer Selbstpruefung, die ohne den COLRv1-Knotendeckel nachweislich anschlaegt
BSP	Beispiele und Erreichbarkeit	4	0	100 %	Fertig (M1491). Der Beispiel-Katalog (DevDoc-PDF und Pruefstand-Startseite) hat 374 Eintraege, Photo und MusicOcr sind erreichbar, und ein Waechter verlangt je Themenbereich genug Beispiele fuer seine Groesse — ein Bereich kann nicht mehr unbemerkt duenn bleiben
EXT	extern	4	0	—	toolchain-/zertifikatsgebunden, kein fehlender Code

Dreamweaver hat keine eigene Gruppe: der HTML/CSS->PDF-Weg ist fertig (LHT, M1468), die Oberflaeche (Code-Editor mit Live-Vorschau /htmleditor, Site-Verwaltung, HTML-Vervollstaendigung/-Validierung) steht seit ED50/ED51 (M1372-Nachtrag), die Vervollstaendigung und Validierung fuer HTML und CSS seit ED52.

MG · Gemeldete Maengel an bereits Gebautem

LVS25 · Datenquellen-Zeichnungen ohne Nachbearbeitung (Nutzermeldung 2026-10-16) — 19 von 20

Ein Werkzeug des Nutzers baute Datenquellen-Zeichnungen mit model_to_vsdx und besserte danach rund 250 Zeilen XML von Hand nach. Seit M1501 kann das Modell alles selbst (LVS25.1–25.19, im Archiv): ein Aufruf je Blatt, danach vsdx_check. Offen ist nur das Ziehen eines Linienendes in Visio:

ID	Wunsch	Stand	Aufw.
LVS25.20	Ein Linienende in Visio am Objekt entlang ziehen (Nutzermeldung): beim Ziehen sprang das Ende an den Seitenrand, statt an der Form weiterzugleiten. Warum noch nicht gebaut: ohne Visio ist nicht belegbar, welche Zellen Visio dafuer erwartet (ein echter „Dynamischer Verbinder“ mit Routing-Zellen, eine gewoehnliche 1-D-Linie oder die heutige Form). Was es braucht: die Rueckmeldung des Nutzers zu vier Probedateien (Varianten A–D: heute, ohne ObjType, mit ShapeRouteStyle/ConLineRouteExt/ConFixedCode, ohne Geometrie-Formeln) — die Variante, die sich richtig ziehen laesst, wird die Schreibweise	fehlt	S

Reihenfolge, die ich empfehle

Nicht nach Aufwand sortiert, sondern nach Nutzen je Aufwand.

Stand M1502. Erledigt seit M1491: MUS16 (Notenblatt-Erkennung — Takteinteilung, Taktart, Auftakt, Mehrtaktpausen, Akkorde, zwei Stimmen,

Boegen), LR7b (Bereichsreparatur und Masken-Reste), IMG7 (AVIF schreiben, 10 Bit fuer HEIC und AVIF), FCR20 (Ueberlappungen entfernen auf Bezierkurven), LR7c (Masken-Gruppen aus Lightroom 11+), LR7e (Lightrooms uebrige lokale Regler), LVS25.1-25.19 (Datenquellen-Zeichnungen ohne Nachbearbeitung; LVS25.20 wartet auf eine Visio-Rueckmeldung des Nutzers), ED43b (Fliesstext-Modus vollstaendig, auch auf der Leinwand).

ROB6 ist erledigt (M1506) — die Rangliste ist leer und wird beim naechsten Planungsschritt neu bewertet. Offen sind die Restzeilen der Gruppen in der Uebersicht oben.

Nicht in der Liste, mit Absicht: LPP10 (Diagramm-Ausschmueckung, SmartArt ohne mitgelieferte Zeichnung — der Rest des in M1474 geschlossenen LPP8) und EXT sind entweder Nischen, Medium-Grenzen (Bewegung in einem statischen PDF) oder nicht abbildbar — sie stehen in ihren Zeilen mit Begruendung und melden sich zur Laufzeit.

v4-AP · Acrobat Pro

Nichts mehr offen. Die erledigten Punkte samt ihren ehrlich benannten Grenzen stehen in roadmap-erledigt-v4.md.

v4-LXL · Excel (.xlsx)

Nichts mehr offen. Die erledigten Punkte samt ihren ehrlich benannten Grenzen stehen in roadmap-erledigt-v4.md.

v4-LFO · Apache FOP (XSL-FO)

Nichts mehr offen. Die erledigten Punkte samt ihren ehrlich benannten Grenzen stehen in roadmap-erledigt-v4.md.

v4-LPP · PowerPoint (.pptx)

Schon da: positionierte Text-/Bildrahmen lesen+schreiben (Folienmasse, Absaetze/Laeufe mit Fett/Kursiv/Groesse/ Farbe/Ausrichtung, Bilder aus ppt/media), pptx->pdf, pdf->pptx (M1205); dazu seit LPP5-LPP9 Notizen/Kommentare, Themenfarben+-schrift, Bild-Zuschnitt/Gruppen/Drehung/Spiegelung, lesende Diagramme (Balken/Linie/Torte) und der rohe Uebergangs-/Zeitplan-Rundlauf. Der Leser kennt heute sp/pic/grpSp/graphicFrame+c:chart/txBody - Layouts/Master-Platzhalter, Tabellen und Formen-Geometrie (prstGeom) sind unten im Code geprueft NICHT vorhanden (LPP1-4).

#	PowerPoint-Funktion	Stand bei dir	Aufw.
LPP11	Restposten nach LPP10: SmartArt ohne mitgelieferte Zeichnung wird weiterhin nicht nachgerechnet — dgm:layoutDef ist eine eigene Beschreibungssprache mit Algorithmen, Randbedingungen und Fallunterscheidungen; sie nach dem Layoutnamen nachzuahmen hiesse, Groessen, Abstaende und Verbindungen zu erfinden, und eine falsch angeordnete Grafik ist schlechter als eine benannte Luecke. Seit LPP10 geht wenigstens der Inhalt nicht mehr verloren (die Knotentexte kommen an). Dazu offen: die c:spPr-Feinheiten der Reihen-Untervariante und manualLayout	fehlt	M

v4-IMG · Bildformate — Lesen und Schreiben

Schon da beim Lesen (je eigener Codec, kein Fremdpaket): JPEG, PNG, TIFF (unkomprimiert/LZW/Deflate/PackBits/ CCITT G3+G4/JPEG-in-TIFF, 1–16 Bit), GIF, WebP (VP8L und VP8), HEIF/HEIC (eigener HEVC-Intra-Decoder, M1147), JPEG 2000 (EBCOT/DWT), JBIG2, CCITT. Geschrieben wurde bei Beginn dieser Gruppe nur PNG (indiziert/TrueColor), JPEG und 1-Bit-TIFF (unkomprimiert, fuer CTP) — genau das war die Luecke: die Lese-Seite war deutlich breiter als die Schreib-Seite. Seither ist sie geschlossen (siehe die Matrix); offen ist nur noch der XL-Codec JPEG XL (HEIF und AVIF schreibt die Bibliothek seit M1496).

Die Matrix — welches Format in welche Richtung (im Code geprueft):

Format	Lesen	Schreiben	Bemerkung
AVIF	Bildpunkte, bit-genau gegen dav1d - verlustfrei UND verlustbehaftet (M1310-M1413), dazu 10 und 12 Bit (M1496)	geschrieben seit M1496 (eigener AV1-Intra-Codierer, 8/10/12 Bit, verlustfrei bit-genau, libavif liest)	Lesen nicht allgemein tragbar: libavif (der Kodierer hinter Chrome, avifenc, Pillow, ImageMagick) schaltet bei jeder verlustbehafteten Stufe die Quantisierungsmatrizen ein, die hier nicht hinterlegt sind — in M1496 an libavif 1.4.2 nachgemessen (Qualitaet 50/75/90: jedes Mal using_qmatrix). Deshalb bleibt .avif aus DocumentConverter draussen (sonst genau der IMG1-Fehler); was dafuer fehlt, ist die normative Tabelle Quantizer_Matrix im Leser, nicht der Schreiber

Kassensturz (aus dem Code gelesen): bei M1306 waren von 15 lesbaren Rasterformaten 6 schreibbar (PNG, JPEG, TIFF, GIF, WebP verlustfrei, JPEG 2000 verlustfrei). Heute sind es alle bis auf JPEG XL und DNG — und DNG ist bewusst nicht vorgesehen (eine Rohdatei zu schreiben hiesse, Sensordaten zu erfinden). Die folgenden Punkte fuehren die Gruppe zu Ende.

#	Bildformat-Funktion	Stand bei dir	Aufw.
IMG8 4/6	JPEG XL lesen + schreiben	Untertabelle unten: IMG8.1–8.6	XL

Detailtiefe je Format. Ein Format „koennen" heisst nicht, jede seiner Spielarten zu koennen. Der folgende Durchgang ist am Code gemessen und nennt, was je Format noch fehlt — meist Kompressions- oder Struktur-Varianten, die in echten Dateien vorkommen.

#	Was fehlt	Warum es vorkommt	Aufw.
IMG29 6/7	JPEG: progressiv schreiben; 12 Bit und arithmetische Kodierung lesen	Untertabelle unten: IMG29.1–29.7	M–L

Die grossen Bildformat-Punkte im Einzelnen

Ein Punkt mit mehr als ein paar Arbeitsschritten steht hier als Gruppe mit nummerierten Unterpunkten - sonst verbirgt ein einzelnes genau das, was man wissen will: wie weit es ist und was noch fehlt. Jede Scheibe ist zugleich der Schnitt fuer ein Agenten-Paket.

IMG8 · JPEG XL — 4 von 6

#	Scheibe	Stand
IMG8.5	VarDCT-Pfad (XYB, variable DCT, Gabor/EPF)	
IMG8.6	Schreiben (modular, verlustfrei)	

IMG29 · JPEG: die Randfaelle — 6 von 7

#	Scheibe	Stand
IMG29.6	SOF11 — verlustfrei arithmetisch	- bleibt benannt abgelehnt, und der Grund ist gemessen, nicht vermutet (M1408)

IMG32 · JPEG 2000: das volle Codestream-Profil — 11 von 11

v4-FNT · Schriften — alle Arten, beide Richtungen

Nichts mehr offen. Die erledigten Punkte samt ihren ehrlich benannten Grenzen stehen in roadmap-erledigt-v4.md.

v4-FED · Schriften — jede Art lesen, schreiben und ineinander wandeln, plus Editor

Der Wunsch: moeglichst alle Schriftarten, die es gibt und die benutzt werden — Web, Druck und Betriebssysteme — lesen, schreiben und in jede andere Art wandeln, ohne Fremdprodukt (kein FontForge, kein fonttools). Dazu ein Editor, um bestehende Schriften zu aendern und neue zu bauen.

Was schon da ist (nicht doppelt bauen): Leser fuer TrueType(glyf), OpenType/CFF, bare CFF, CID-keyed CFF, CFF2, Type1(PFB/eexec), Type3, Type42, WOFF, WOFF2, Variable Fonts (inkl. Instanzieren) und Farbfonts COLR/CPAL v0+v1; Umriss-Auslesung (GlyphBoundsReader), sfnt-Schreiber (SfntWriter, SfntTableBuilders), CFF->OpenType-Verpackung, Subsetter fuer TrueType und CFF, Type1FontConverter (-> PFB/PFA), Cff2StaticFontBuilder und FontExporter (M990), der eine eingebettete Schrift bereits als .ttf/.otf/.pfb herausschreibt und fehlende Tabellen ergaenzt.

Die Punkte

Ehrlich zur Verifikation: der ueberzeugende Beleg fuer eine selbst gebaute Schrift ist, dass ein fremder Renderer sie liest (HarfBuzz/FreeType, wie schon bei den OpenType-Attributen M1087). Nach der Projektregel geschieht das einmalig beim Bauen; im Repo bleibt die eingebackene Erwartung.

Ehrlich zur Vollstaendigkeit: „alle Schriftarten" heisst hier alle, die im Web, im Druck und in Betriebssystemen tatsaechlich vorkommen — die Matrix oben zaehlt sie auf. Firmeneigene Editor-Formate (FontLab VFB/VFC, Glyphs .glyphs) sind nicht dokumentiert; sie liessen sich nur durch Nachbau

aus Beispieldateien erschliessen und stehen bewusst nicht in der Liste.

v4-FCR · Schrift-Erzeuger und -Editor (Nutzerwunsch)

„ich moechte einen echten font creator/editor haben mit der darstellung des editors"

Zuerst, was es schon gibt und was nicht — sonst klingt dieser Abschnitt nach einer Luecke, die keine ist. Die Rechenkerne stehen vollstaendig (v4-FED, 20/20): Umrisse lesen, aendern und zurueckschreiben (glyf und CFF-Charstrings), eine Schrift von Grund auf bauen, Metriken und Kerning, cmap, Glyphennamen, variable Achsen (gvar und CFF2), Farbschriften in allen drei Arten, und die Ausgabe in jedes Format der Matrix. Was fehlt, ist das Werkzeug darueber: /schrifteditor (FED7) ist eine Formularseite, auf der sich eine Glyphe waehlen, ein Umrisspunkt ziehen und die Vorbreite setzen laesst. Das ist kein Schrift-Editor, und es sollte auch nie einer sein — es war der Nachweis, dass die Kerne von aussen bedienbar sind.

Der Massstab ist ausdruecklich der grosse Editor, nicht eine Formularseite: Werkzeugleiste, Eigenschaften-Feld an der Seite, freie Arbeitsflaeche mit Linealen, Rueckgaengig-Kette, dasselbe dunkle Thema. Vorbilder sind FontForge, Glyphs und RoboFont.

Die Bauregel aus v4-ED gilt hier von Anfang an (dort wurde sie erst nach 7170 Zeilen Editor.razor erzwungen): Rechenlogik in eigene Playground/Model/-Dateien als reine, testbare Funktionen, Oberflaeche in eigene Komponenten — in der Seite selbst steht nur die Einbindung.

#	Punkt	Stand	Aufw.

Ehrlich benannt, was hier NICHT hineingehoert. Ein Hint-Editor (Punkte an Zonen binden, Delta-Anweisungen setzen) ist ein eigener Schritt: der Bytecode-Interpreter steht (FNT4, 122 Instruktionen), das Schreiben von Hinting-Programmen von Hand ist etwas anderes. Ebenso das automatische Hinting und die automatische Zeichenabstand-Ermittlung — beides sind eigene Verfahren, keine Oberflaechenarbeit. Und ein Zeichnen mit dem Stift auf dem Tablett setzt Druck- und Neigungsdaten voraus, die eine Blazor-Server-Oberflaeche nicht bekommt.

Die Reihenfolge ist erzwungen, nicht gewaehlt: FCR1 und FCR2 tragen alles Uebrige — ohne Arbeitsflaeche mit den richtigen Hilfslinien laesst sich kein Punkt sinnvoll setzen. FCR3 und FCR4 gehoeren zusammen (ein Zeichenwerkzeug ohne Stetigkeit erzeugt nur Knicke). FCR13 und FCR14 setzen FCR3 voraus, FCR7 setzt FCR10 voraus.

v4-TXT · Klartext / Notepad

Nichts mehr offen. Die erledigten Punkte samt ihren ehrlich benannten Grenzen stehen in roadmap-erledigt-v4.md.

v4-AG · Apogee-Rest (Druckvorstufe)

Nichts mehr offen. Die erledigten Punkte samt ihren ehrlich benannten Grenzen stehen in roadmap-erledigt-v4.md.

v4-KDX · .kdoc als Vollabbild jedes Formats (Nutzerwunsch)

Ziel: Import -> .kdoc -> Export ins selbe Format verliert nichts. Befund und Entwurf in docs/superpowers/specs/2026-09-19-kdoc-vollabbild.md (vier Analysen am Code). Drei Wege: direkte Import-/Exportwege statt PDF, Modell erweitern, Originalteile-Speicher fuer das, was das Modell nicht abbildet.

Nichts mehr offen. Die erledigten Punkte samt ihren ehrlich benannten Grenzen stehen in roadmap-erledigt-v4.md.

v4-KMU · Musik im .kdoc — mehr als MusicXML und MIDI (Nutzerwunsch)

Nutzerwunsch woertlich: *„kdoc ist doch frei definierbar und es kann nur das rein holen was auch vorhanden ist, aber danach laesst sich alles noch anreichern ueber den Editor … ich moechte das kdoc so ausbauen das es auch in dem Bereich mehr kann als die anderen Formate, so das sicher alles damit abdeckbar ist was in der Musik moeglich ist.“

Befund und Schnitt in docs/superpowers/specs/2026-09-24-musik-im-kdoc.md (am Code gemessen). Der Zuschnitt folgt dem Excel-Muster aus M1470: ein .kdoc-Datenteil als Bestand, aus dem MusicXML und MIDI zurueckgeschrieben werden, und die gesetzten Seiten als Ansicht ueber genau den vorhandenen ScoreEngraver/MusicPdf — kein zweiter Notensatz.

Die Reihenfolge ist erzwungen, nicht gewaehlt: KMU1 ist der Schluesselstein (ohne Datenteil gibt es nichts zu speichern), KMU2 traegt den Import, und jede Modell-Scheibe ab KMU4 muss Modell und Stich in einem Zug bringen — sonst entsteht genau der Fall, der in diesem Projekt schon dreimal aufgetreten ist: ein Feld, das gespeichert wird und das beim Zeichnen niemand ausliest.

#	Punkt	Stand bei dir	Aufw.
KMU7b	Tabulatur-Restposten nach KMU7: eine Pause in einer N-tole bekommt keine eigene Klammer-Kante; Text und Ausdruck in einer Tabulatur-Zeile werden nicht gesondert gesetzt. Ein Fund aus dem Orakel, festgehalten weil er wiederkehren wird: <staff-type>tablature</staff-type> gibt es nicht — music21 warnte, das Schema kennt dort nur ossia/editorial/cue/alternate/regular; die Tabulatur sagen allein sign=TAB und die staff-tuning-Eintraege. Und das Format zaehlt gemischt: staff-tuning line="1" ist die unterste Linie, technical/string 1 die hoechste Saite (belegt an der Unofficial MusicXML Test Suite, 18 Noten)	fehlt	S
KMU13	Historische Notation: Mensuralnotation und Neumen als eigene Notenzeilen-Art. Vorab benannt: das ist kein Nachtrag am vorhandenen Stich, sondern ein eigenes Zeichenwerk — Mensuralnotation kennt weder Taktstrich noch Notenhals im heutigen Sinn	fehlt	XL
KMU21b	N-tolen-Restposten nach KMU21a: eine N-tole ueber einen Zeilenumbruch (die zweite Haelfte steht in einem System mit eigener Geometrie); die N-tole verbreitert die Spalte nicht; die Klammer weicht nichts aus (Hilfslinien, Beischriften, zweite Stimme); eine Duole/Quartole in zusammengesetzter Taktart bekommt das volle Verhaeltnis statt nur der Zahl. Je zur Laufzeit gemeldet	fehlt	M
KMU22b	Restposten nach KMU22a: die Textbreiten von Volten-Ziffer, Anweisung und Probenbuchstaben-Rahmen sind genaehert; Beischriften weichen einander nicht aus; der durchgehende Gruppen-Taktstrich bleibt ein duenner Strich, auch wenn die Gruppe eine Wiederholung traegt; der gepunktete Taktstrich wird kurz gestrichelt gezeichnet, weil die Leinwand keine runden Linienkappen kennt	fehlt	M

#	Punkt	Stand bei dir	Aufw.
KMU24	Restposten nach KMU10: die gedruckten Taktzahlen gehen nicht durch MusicXML — sie sind eine Stech-Regel des Dokuments (EngravingRules), MusicXMLs <measure-numbering> haengt dagegen am Takt; es braucht eine Abbildung, die beim Zuruecklesen wieder eine Dokumentregel ergibt. Gedruckt wird Measure.Number, wie es dasteht — gezaehlt wird nicht; eine Zaehlung muesste Auftakt, mehrtaktige Pause (eine Nummer vertritt mehrere Takte) und Wiederholungen beruecksichtigen und liegt damit naeher bei KMU12. Ein verstecktes Element verbraucht weiter seine volle Breite (MusicXMLs print-spacing="no" fehlt); Part.StaffSizeFactor skaliert den Notenzeilen-, nicht den Systemabstand und nicht die Stimmennamen; SystemDistanceBefore ist bewusst additiv zu LineGap und wirkt nicht, wenn die Zeile ohnehin oben auf der Seite steht	fehlt	M
KMU25	Restposten nach KMU18 (Instrumentenwechsel): die Spielweisen-Pruefung arbeitet seit KMU8 am Modell (Flageolett, Daempfer, Tremolo und Bogenstrich stehen dort jetzt); die uebrigen Anweisungen erkennt sie ueber eine kurze Wortliste in MusicDirection.Text, und ein nicht erkannter Text wird nicht gemeldet (lieber eine Anweisung uebersehen als eine gewoehnliche Vortragsangabe faelschlich als unpassende Technik ausgeben). Die Oktavversetzung wirkt je Zeitstelle, nicht je Phrase — ein Bearbeiter haette den ganzen Bogen versetzt, hier kann ein einzelner Ton springen. Die Familie geht nicht durch MusicXML (das Format hat keinen Begriff dafuer; score-instrument nennt den Klang, nicht die Familie) und ueberlebt nur ueber das .kdoc. Die Umfangspruefung kennt die Tonhoehe, nicht die Spielbarkeit eines Griffs oder Sprungs. Dazu fehlt das Pruefstand-Beispiel fuer transcribe_part	fehlt	M
KMU26	Restposten nach KMU12 (Wiedergabe): Vorschlagsnoten werden nicht gespielt und das wird gemeldet — sie haben im Modell keine eigene Zeit, und ihnen welche zu geben hiesse die folgende Note zu verschieben; wie viel ist eine Auffuehrungsentscheidung. Artikulation und Verzierungen wirken nicht (Staccato kuerzt nicht, ein Triller wird nicht ausgespielt); sound dynamics wird weiter gemeldet statt gelesen (eine Anschlagstaerke in Prozent laesst sich nicht verlustfrei auf die gedruckten Stufen pp..ff abbilden). Die Wiederholungsstruktur kommt aus der ersten Stimme, eine abweichende wird gemeldet; RepeatUnfolding kennt ein Segno, eine Coda, und eine Sprunganweisung wirkt genau einmal. Im MIDI gibt es keinen Programmwechsel (welches Klangprogramm zu „Viola" gehoert, entscheidet der Klangerzeuger), Schlagzeug liegt seit KMU6 wirklich auf Kanal 10, aber ab der 16. Stimme teilen sich Stimmen einen Kanal; die Tempokurve ist eine Treppe aus 8 Stufen	fehlt	M
KMU27	Restposten nach KMU6 und KMU9: Part.StaffLines gilt fuer alle Notenzeilen einer Stimme, nicht je Zeile — ExtraStaves ist heute eine blosse Taktliste ohne Darstellungsangaben, eine Zahl je Zeile braeuchte eine eigene Angabe daneben. Keine Kopfform je Akkordton: ohne die dazugehoerige Stechregel (welcher Kopf wohin ausweicht) verspraeche sie mehr, als der Satz einloest. Die Viertelton-Zeichen folgen nur Stein/Zimmermann — SMuFL fuehrt Alternativen (Pfeile, Couper, Wyschnegradsky); eine davon zu waehlen hiesse, eine Schule zu bevorzugen, ihre MusicXML-Namen werden gelesen, ihre abweichende Zeichnung geht verloren. Und die Stichnote ist nicht fremdbelegt: music21 liest <cue/>, gibt sie aber weder ueber style.noteSize noch duration.isGrace aus (gemessen) — dort traegt nur der eigene Rundlauf	fehlt	M
KMU29	Restposten der Editor-Bedienung (nach KMU20): je Zeitstelle laesst sich eines statt mehrerer setzen — eine Verzierung, eine Spieltechnik, eine Spielanweisung, eine Sprungmarke je Takt, eine N-tolen-Ebene, eine Generalbass-Zeile, eine Liedtext-Strophe. Ein Taktmasswechsel mitten im Stueck fehlt (Schluessel und Tonart sind einstellbar) — er aenderte die Takteinteilung selbst. Taktangaben an einem Eintrag der zweiten Stimme gehen verloren, weil der Bau von deren Takten nur die Noten uebernimmt. Und die vierte Notenzeile wird weiter gekappt (MusicTrack.MaxNotenzeilen = 3): der Notenstich hat gemessen keine Obergrenze, und der Instrumentenkatalog fuehrt kein Instrument ueber drei Zeilen — eine fremde MusicXML mit vier Systemen verliert das vierte; anheben waere plausibel, ist aber ungetestet und wurde deshalb nicht geraten. Kein Mangel ist dagegen die dritte Stimme: MaxStimmen = 2 folgt dem Notenstich, der nur zweien die Halsrichtung nach der Rolle gibt — eine dritte waere falsch gesetzt, nicht bloss fehlend	fehlt	M
KMU30	Restposten der Eingabezeile (nach KMU19): sie hat weiterhin keinen Zeilenumbruch (ein System), setzt alle Takte gleich breit, und die Tonart wirkt nicht auf die Trefferermittlung — ein Klick im vorgezeichneten fis-Bereich liefert f (die Versetzung setzt der Nutzer, sie wird nie geraten). Gestaltung erst nach dem Profi-Vorbild, das der Nutzer waehlt	fehlt	M

v4-ED · Editor — Paritaet zu den Vorbildern

Der Editor selbst (nicht die Bibliothek): was die Oberflaeche gegenueber InDesign, Acrobat Pro, Word, Excel, PowerPoint, Dreamweaver und Notepad noch WENIGER kann. Im Code der Editor-Seite geprueft (Editor.razor 7170 Zeilen + EditorService.cs 4681 Zeilen), nicht aus dem Gedaechtnis. Schon da (NICHT gelistet): WYSIWYG-Canvas mit Live-Darstellung, Inline-Textbearbeitung, Undo/Redo, Menue-/Kontroll-/Statusleiste, Rechtsklick-Menue, Musterseiten + Doppelseiten + Miniaturen, Lineal-Hilfslinien, Montageflaeche, Ebenen/OCG, Farbfelder mit eigenem Farbwaehler, Verlaeufe, Absatzformate, Verknuepfungs-Palette, Formularfeld-Bedienung, Lesezeichen, Tabellen mit Zelltext/-farben, Drehung, Suchen+Ersetzen, Rechtschreibpruefung auf Knopfdruck, Seiten-Organisierer mit Drag&Drop, Dokument-Import/-Export, PDF-Export-Dialog.

Nichts mehr offen (M1502). Die erledigten Punkte samt ihren ehrlich benannten Grenzen stehen in roadmap-erledigt-v4.md.

v4-ARC · Archive — Rest

Alle Formate sind seit M1433 lesbar (siehe roadmap-erledigt-v4.md). Offen sind die mehrteiligen Archive und die Grenzen, die M1433 ehrlich benannt hat.

#	Punkt	Stand bei dir	Aufw.
ARC22	Apple Archive schreiben (blank und im pbz*-Mantel; LZFSE-Packer)	nur gelesen	M
ARC27	StuffIt vervollstaendigen: die Methoden 5/6/8/14, Verschluesselung und StuffIt X lesen, dazu StuffIt schreiben (unar/lsar taugen als Gegenpruefung)	Methoden 0/1/2/3/13/15 lesen	M
ARC28	.dmg schreiben — UDIF-Behaelter mit einem HFS+-Schreiber darin	nur gelesen	L
ARC29	APFS verschluesselt (mit Passwort/Schluessel), versiegelte Volumes, LZBITMAP, bvx1; Fusion als Sonderfall	benannt abgelehnt	L
ARC30	Stromweiser Weg fuer Archive, die nicht in den Speicher passen — durch alle Leser und Schreiber	jedes Archiv liegt ganz im Speicher	L
ARC31	StuffIt-Konstanten aus unar (LGPL): die festen Codelaengen der Methode 13 und die Arsenic-Tabelle — rechtliche Einschaetzung durch den Nutzer, sonst unabhaengig herleiten	Entscheidung offen	S
ARC33b	RAR-Restposten nach ARC33a: die RAR4-Schluesselableitung (sie schnappschusst den SHA-1-Zustand alle 0x4000 Runden, und .NETs IncrementalHash laesst sich nicht klonen — es braucht eine eigene SHA-1); die Wiederherstellungsdaten; und „verschluesselt UND solid“, das im Verbund weiterhin benannt abgelehnt wird. Fuer alle drei fehlt das Orakel, und das ist kein Versaeumnis: WinRAR 7.22 erzeugt kein RAR4 mehr (-ma4 ist unbekannt), es gibt also kein verschluesseltes RAR4 zum Gegenlesen — die Ableitung ohne Pruefmoeglichkeit aus dem Gedaechtnis nachzubauen waere Raten. Ebenso bleibt RAR4 mehrteilig ungeprueft: der Code nimmt denselben Weg wie RAR5, belegt ist er nur fuer RAR5	fehlt	M
ARC36b	ZIP-Verbund ueber 4 GB vereinigen — und nicht wegen ZIP64 (das geht seit M1484): das Ergebnis wird als ein byte[] gehalten und scheitert schon bei 2 GB an Array.MaxLength. Das braucht den stromweisen Weg aus ARC30	fehlt	S

v4-MUS · Notensatz — Notenblaetter erstellen, mit eigenem Editor (Nutzerwunsch)

Ziel: Notenblaetter mit allem, was dazugehoert, selbst setzen — und einen eigenen Editor dafuer. Vorbilder sind Sibelius/Finale/Dorico und das freie MuseScore; der Austausch laeuft ueber MusicXML (das Format, das alle koennen) und MIDI.

Was dafuer schon da ist (nicht doppelt bauen): das Vektor-Zeichnen (PageCanvas), das Objekt-Modell mit Seiten (LayoutDocument) und sein .kdoc-Format, die Schrift-Einbettung samt Umriss-Auslesung und OpenType-Shaping (Notenschriften wie Bravura sind gewoehnliche OpenType-Schriften nach dem SMuFL-Standard), der abgesicherte XML-Leser (fuer MusicXML) und System.IO.Compression (fuer das gepackte .mxl), dazu die ganze Editor-Oberflaeche (Canvas, Auswahl, Undo/Redo, Paletten), auf der ein Noten-Editor aufsetzen kann.

#	Funktion	Stand bei dir	Aufw.
MUS4	Umbruch: Takte auf Systeme, Systeme auf Seiten, Zeilenausgleich, Wiederholungen und Sprungmarken (D.C./D.S./Coda), Taktzahlen	Music/ScoreLayout + ScoreAssembly (M1404): Zeilen- und Seitenumbruch in zwei Durchlaeufen - erst jeden Takt probeweise setzen, um seine natuerliche Breite zu messen (kein zweiter Rechenweg), daraus Spaltenbreiten als Maximum ueber alle Stimmen, sodass die taktweise Ausrichtung zwangslaeufig entsteht; gedehnt wird nur der Abstand zwischen den Zeitstellen, nie die Geometrie einer Note. Die letzte Zeile bleibt ungedehnt (Stichregel). Taktzahlen werden seit KMU10 (M1488) gedruckt (Measure.Number ueber dem Taktanfang). Offen: Wiederholungen und Sprungmarken beim Umbruch, Seitenkopf/-fuss; der Umbruch ist greedy, kein Knuth-Plass-Optimum	L
MUS8	Sonderfaelle: Schlagzeug-Notation, Gitarren-Tabulatur mit Griffbildern, Prozent-Wiederholungen, Cue-Noten, mehrtaktige Pausen	Mehrtaktige Pausen: MusicRest.MultiMeasureCount haelt eine ganze Reihe leerer Takte als EIN Element; der Notenstich zeichnet dafuer einen dicken Balken mit Serifen (Geometrie, kein SMuFL-Glyph - steht damit auch ohne Notenschrift da) samt der Taktzahl fett darueber. PartExtractor.CollapseRests (MUS7) erkennt zusammenhaengende leere Takte automatisch und fasst sie zusammen - ein Takt mit einer Note ODER einem eigenen Schluessel-/Tonart-/Taktart-/Tempowechsel bleibt eigener Anker, damit dabei nichts verloren geht; die gedruckten Taktnummern bleiben dabei die echten der Partitur (sie muessen nicht lueckenlos sein). Gitarren-Tabulatur mit Griffbildern ist mit KMU7 (M1489) erledigt. Offen: Schlagzeug-Notation, Prozent-Wiederholungen, Cue-Noten	M
MUS12b	SVG traegt den Notensatz nicht — gemessen: LayoutSvgWriter uebersetzt eine feste Reihe von Rahmenarten und uebergeht einen MusicFrame; heraus kommt ein gueltiges, leeres SVG. Ihn dort nachzubauen hiesse, auch die Zeichen der Notenschrift als Umrisse zu setzen. Ein eigener Test haelt den heutigen Zustand fest, damit es auffaellt, sobald es jemand schliesst. Ebenso kommt der Titelblock von MusicPdf.Render ueber ScoreExport nicht mit	fehlt	M

#	Funktion	Stand bei dir	Aufw.
MUS13 16/25	Der Noten-Editor — Untertabelle unten: MUS13.1–13.22 (Nutzerwunsch: „so das er freistehend sein kann wie der editor und auch sonst alles naeher am editor")	Pruefstand-Seite /noteneditor mit eigenem Knopf: Spuren mit Instrument (39), Systemgruppen, Notenblatt-Definition, Umschalter klingend/notiert, Vorschau und PDF - seit M1421 dazu die Eingabezeile: klicken setzt, ziehen waehlt aus, Pfeiltasten und Buchstaben setzen und verschieben, dazu Werkzeugleiste, Rueckgaengig-Kette und Speichern/Laden. Die Logik liegt in reinen, testbaren Funktionen unter Playground/Model/, die Razor-Seite ruft nur auf. Seit KMU19/KMU20 (M1488) bedient die Oberflaeche Akkorde, zwei Stimmen, beide Klavierzeilen, Dynamik, Artikulation, Binde- und Haltebogen, Balken, Gesangstext, Akkordsymbol, Tempo sowie Schluessel- und Tonartwechsel je Takt. Offen: Noten zwischen zwei Notenzeilen verschieben, der Taktartwechsel mittendrin, ueberlange Noten binden, die Takt-Fuellung korrigieren, die Tastatur-Zuordnung als Oberflaeche und Vorgaben nach Besetzung	XL
MUS17	Restposten des Anhoerens (nach MUS14): es klingt ein Tongenerator, kein Instrument — keine Klangfarben; Vorschlagsnoten, Artikulation und Verzierungen wirken nicht auf den Klang (so sagt es die Wiedergabeschicht selbst, KMU26), das Pedal wird nicht in laengere Toene umgesetzt, und waehrend der Wiedergabe gibt es keine Positionsanzeige in der Partitur. Ein echter Synthesizer (SoundFont/SF2 oder ein eigener) ist ein eigener Brocken und wurde bewusst nicht angefangen	fehlt	L

Ehrlich vorweg gesagt: MUS3 (die feinen Stichregeln) war der eigentliche Brocken und ist in M1426 abgeschlossen: Balkenraster (Sit/Straddle/Hang), melodische Konturneigung, Kollisionsausweichen fuer Bindeboegen, Fermaten und Beischriften sowie elastische Spaltenbreitenkompensation fuer gestaffelte Vorzeichen und Text.

Zur Verifikation: fuer eingelesenes MusicXML ist music21 (Python) ein unabhaengiger Leser, und MuseScore/LilyPond koennen dasselbe Stueck zum Vergleich setzen. Nach der Projektregel geschieht das einmalig beim Bauen; im Repo bleibt die eingebackene Erwartung, kein Fremdwerkzeug-Aufruf.

MUS13 · Der Noten-Editor im Einzelnen — 9 von 22

Nutzerwunsch (woertlich): „ich moechte auch den noten editor so das er freistehend sein kann wie der editor und auch sonst alles naeher am editor so das man auch per drag drop oder per selektion und an der gewuenschten position die noten oder sonstige sachen plazieren kann und auch noten verbinden mit punkt leerem kreis noetenschluessel und all dem anderen was noetig ist. ich moechte auch ein klavier das unten ein und ausgeblendet werden kann mit dem man die noten selber eingeben kann und dazu die maus oder die tastatur verwenden kann mit einer tastatur mapping funktion in den settings."

#	Scheibe	Stand
MUS13.7	Noten mit der Maus setzen: klicken an der gewuenschten Stelle, Tonhoehe aus der Zeile, Zeitstelle aus der Waagerechten	M1421 - MusicHitTest kehrt die Stech-Geometrie um (Bildpunkt -> Stufe ueber den Schluessel, X -> Takt und Zeitstelle), MusicEingabezeile zeichnet mit derselben Geometrie. Zwei ehrlich benannte Grenzen: getroffen wird das gleichmaessige Eingaberaster, nicht das gesetzte Bild (aus dem liesse sich nicht zurueckrechnen, welcher Bildpunkt zu welchem Eintrag gehoert); und der Klick bestimmt die Stelle in der Folge, nicht eine beliebige Zeitstelle - eine Luecke davor wird nicht mit Pausen gefuellt
MUS13.9	Paletten zum Anklicken: Notenwerte (ganz/halb/viertel… als leerer und gefuellter Kopf), Punktierung, Pausen, Notenschluessel, Vorzeichen, Taktangabe	Notenwerte, Punktierung, Pausen und Vorzeichen sind bedienbar. Offen: der Schluessel kommt heute nur ueber das Instrument der Spur, die Taktangabe nur ueber die Notenblatt-Definition - beides nicht als Palette
MUS13.10b	Noten verbinden mit der Maus: einen Bogen im Notenbild ziehen statt ihn ueber ein Auswahlfeld zu setzen, und eine Balkengruppe als Ganzes loesen oder neu bilden	- heute je Eintrag ueber ein Auswahlfeld (13.10a)
MUS13.13	Tastatur-Zuordnung in den Einstellungen — frei belegbar und gespeichert	M1421 - der Rechenkern steht (MusicKeymap: frei belegbar, als Text speicher- und ladbar, kaputte Zeilen werden gemeldet und uebersprungen, eine unbrauchbare Eingabe faellt auf die Vorgabe zurueck und auch das wird gemeldet). Offen: die Bedienoberflaeche dafuer - der Editor nutzt heute die Vorgabe-Belegung
MUS13.15b	Ueber zwei Notenzeilen zugleich arbeiten: Auswahl ueber beide Zeilen und eine Note von einer Zeile in die andere verschieben	- heute wird immer genau eine Zeile bearbeitet; ein Klick in die andere wechselt dorthin, statt dort zu setzen
MUS13.19b	Taktartwechsel mittendrin	- er aendert die Takteinteilung selbst; der Editor nennt es unter 'Was dieser Editor nicht kann'
MUS13.20	Ueberlange Note ueber den Taktstrich binden: eine Note, die nicht mehr in den Takt passt, wird aufgeteilt und mit Haltebogen verbunden	- heute beginnt sie den naechsten Takt ganz, der Takt davor bleibt unvollstaendig
MUS13.21	Takt-Fuellung pruefen und korrigieren: nicht genau gefuellte Takte im Editor markieren und auf Wunsch mit Pausen auffuellen	- heute meldet nur der Satzbericht pauschal „ein Takt ist nicht genau gefuellt", ohne Taktnummer und ohne Abhilfe
MUS13.22	Besetzung mit Vorgaben: Systemgruppen (Akkolade/Klammer) nach Besetzung vorschlagen und Balkengruppierung nach Taktart einstellbar machen	- ohne ausdrueckliche Gruppen stehen die Stimmen ungeklammert untereinander; die Balken folgen einer vereinfachten Schlagzaehlung (ein Viertel, bei Achtelnenner drei Achtel) - zusammengesetzte Taktarten wie 7/8 (2+2+3) oder halbtaktige Balken im 4/4 fehlen

Die erledigten Unterpunkte (13.6, 13.8, 13.10a, 13.11, 13.12, 13.14, 13.15a, 13.16–13.18, 13.19a) stehen in roadmap-erledigt-v4.md.

Ehrlich zur Reihenfolge: MUS13.7 ist die Voraussetzung fuer 13.8–13.12 — solange aus einem Mausklick keine Tonhoehe und Zeitstelle wird, laesst

sich nichts setzen, ziehen oder ueber eine Klaviatur eingeben. Und 13.14 kann nicht vor MUS5/MUS6 fertig werden: was der Notensatz nicht zeichnet, kann der Editor nicht anbieten.

v4-PRN · Virtueller PDF-Drucker (Nutzerwunsch)

Auf jedem Geraet „Drucken" waehlen, das Ergebnis entsteht als PDF im Pruefstand und steht dort — mit Warteschlange — wieder zur Verfuegung. Bewusst KEIN Windows-Druckertreiber: der braeuchte ein signiertes Treiberpaket (WHQL/Code-Signing), einen Port-Monitor und liefe nur unter Windows. Der offene Weg ist ein IPP-Drucker (Internet Printing Protocol): Windows, macOS, iOS, Android und Linux drucken ohne jeden Treiber darauf, und moderne Clients senden den Job bereits als PDF.

#	Drucker-Funktion	Stand bei dir	Aufw.
PRN3 7/7	Bekanntmachung im Netz (mDNS/Bonjour _ipp._tcp + AirPrint-Attribute) — der Drucker taucht von selbst in der Geraeteliste auf. Untertabelle unten: PRN3.1–3.7	- Unter Linux GEMESSEN (Debian-Container, .NET 9): auf Port 5353 sitzt dort meist schon ein Responder (avahi/mDNSResponder), doch SocketOptionName.ReuseAddress bildet auf Unix so ab, dass sich unser Socket dazubindet - belegt neben einem Halter mit SO_REUSEADDR (so bindet avahi) und auch benutzeruebergreifend (Halter als root, wir als anderer Benutzer); der Multicast-Beitritt gelingt ebenfalls. Gegenbeleg: haelt jemand den Port ganz ohne Reuse-Option, scheitert das Binden mit AddressAlreadyInUse - dann faengt der Dienst die Ausnahme und laeuft ohne Bekanntmachung weiter (der Pruefstand scheitert nicht). macOS bleibt ungeprueft (keine Maschine dafuer).	M

PRN3 · Der Drucker findet sich von selbst — 7 von 7

Nutzerwunsch (woertlich): „beim spooler moechte ich auch wenn er im docker vorhanden ist das er als drucker im gleichen netzwerk segment erkannt wird und zb durch windows, mac, und linux automatisch den treiber holt und er sich installieren kann."

#	Scheibe	Stand
PRN3.7	End-zu-End-Abnahme je System (Windows, macOS, Linux, iOS, Android)	M1409 - gegen ein echtes CUPS abgenommen (docker/docker-compose.cups-abnahme.yml, scripts/cups-abnahme-starten.sh, Rueckgabewert = Ergebnis): ipptool meldet status-code = successful-ok, lpadmin -m everywhere richtet treiberlos ein, und zwei lp-Auftraege kamen als zwei PDFs an. Die Artefakte liegen als Fixtures im Repo - im Testlauf braucht es weder Docker noch CUPS. Ehrlich offen: Windows, macOS, iOS und Android sind ungeprueft - dafuer fehlt ein zweites Geraet im Netz. Dabei fiel eine Extraktor-Luecke auf: CUPS bettet Type0/Identity-H ohne /ToUnicode ein, PageTextExtractor liefert dann keinen Klartext (der Test gewinnt ihn ueber die cmap der eingebetteten Schrift zurueck)

Was „automatisch den Treiber holen“ wirklich heisst — die Unterscheidung entscheidet ueber den Aufwand: bei IPP Everywhere / AirPrint / Mopria gibt es keinen Treiber. Das Betriebssystem fragt den Drucker nach seinen Faehigkeiten und baut die Warteschlange daraus — genau deshalb ist PRN3.4 der eigentliche Punkt, nicht ein Download. Ein echtes Treiberpaket, das Windows aus dem Netz zieht und installiert, waere etwas anderes:

seit Windows 10 ist das eine Print Support App aus dem Microsoft Store (v4-Treibermodell), also Store-Veroeffentlichung samt Signierung — nicht baubar ohne Store-Konto, und fuer einen virtuellen Drucker im eigenen Netz auch nicht noetig. Diese Grenze steht hier, damit sie nicht spaeter als Ueberraschung auftaucht.

v4-LR · Lightroom (Foto-Entwicklung und Katalog) — Nutzerwunsch

Die Bausteine dafuer sind zu grossen Teilen da: eigene Bild-Codecs (IMG), ICC-Farbmanagement mit echter Umrechnung (Icc), Resampler mit vier Verfahren (M1155), GPU-Kernel fuer Bildpunkt-Arbeit (Accel, M1023/ M1031), Kontaktbogen (PhotoSheetComposer) und der PDF-/Druckweg. Was fehlt, ist die Foto-Entwicklung selbst und der Katalog.

#	Lightroom-Funktion	Stand bei dir	Aufw.
LR1	RAW entwickeln: Demosaicing (Bayer/X-Trans), Weissabgleich als Temperatur/Tint, Kameraprofile	die Grundentwicklung steht mit M1309 (bilinear + Aufnahme-Weissabgleich); es fehlen X-Trans und echte Kameraprofile - Temperatur/Tint als Regler steht seit M1404 (BasicAdjustments.WhiteBalance, in Mired gerechnet); diese Zeile war bis M1480 zu pessimistisch	XL
LR7d	Lightrooms .acr-Raster lesen (KI-Masken, seit Lightroom Classic 15 / Camera Raw neben Bild und XMP): die Masken-Gruppen und KI-Masken selbst werden seit LR7c gelesen, geschrieben und — mit gesetztem Raster — angewandt; die eigene Rasterdatei nicht. Warum nicht gebaut: der Aufbau ist nicht veroeffentlicht, keine frei lizenzierte .acr war auffindbar (GitHub-Code-Suche, Foren), und die einzige Beschreibung (ein Importer-Ticket: Kopf ACR\0, dann JPEG-XL-Raster) nennt weder Byte-Aufbau noch die Zuordnung eines Rasters zu crs:MaskDigest. Was es braucht: eine .acr samt ihrem XMP unter freier Lizenz oder eine Lightroom-Classic-15-Installation, die eine erzeugt; der JPEG-XL-Leser fuer die Raster liegt bereit (Imaging.JpegXlDecoder), die Anwendung eines Rasters ebenso (ImageMask.Raster)	fehlt	S

#	Lightroom-Funktion	Stand bei dir	Aufw.
LR7f	Lightrooms lokales LocalGlow (in der Camera-Raw-18.6-Ausgabe als Wert/100 belegt: Regler +13 als 0.13, photoshop-mcp, MIT): wird beim Lesen uebergangen. Warum nicht gebaut: die Wirkung ist nirgends beschrieben oder gemessen — weder in Adobes Hilfe noch in exiftool, und RAWmakase (MIT), das Camera Raw sonst Regler fuer Regler nachmisst, lehnt Glow ausdruecklich als „nicht gemessen" ab; geraten waere sie ein stiller Fehler. Was es braucht: eine Messung an Camera Raw (Graukeil und Lichtpunkt mit Glow 0/50/100) oder eine Beschreibung von Adobe; danach ein globaler Baustein wie bei LR7e und die Kopplung an LocalAdjustment. Ebenso uebergangen, weil kein Regler dazu gefunden ist: LocalCorrectedDepth, LocalBrightness. Belegsuche 2026-10-11 (Ergebnis: weiterhin nicht belegt, nichts gebaut): Gefunden: Glow kam mit Camera Raw 18.6 als Regler im Effekte-Bedienfeld (unter Koernung) mit Stil Diffusion/Bloom/Halation sowie Bereich, Ausbreitung und Waerme; laut Adobe-Hilfe (Suchauszug, die Seite selbst antwortet 403) sind Stil/Bereich/Ausbreitung/Waerme global und fuer alle Masken gemeinsam, nur der Glow-Betrag ist je Maske einstellbar, lokal auch negativ (nimmt globalen Glow zurueck; Fstoppers, PetaPixel). Die Wirkung ist nur qualitativ beschrieben (Helligkeit und Farbe streuen von hellen Stellen in die Umgebung, Schatten weitgehend unberuehrt). Nicht gefunden: keine Formel, kein Unschaerferadius, keine Schwelle, keine Mischart — weder bei Adobe noch in Fachartikeln; kein frei lizenziertes Projekt, das Glow gemessen hat (RAWmakase fuehrt kein Glow); keine XMP-Namen der globalen Glow-Einstellungen (Adobes crs-Namensraumseite listet sie nicht), keine echte Vorgabe mit LocalGlow ausser der bekannten photoshop-mcp-Ausgabe. Folge: lokal geht Glow erst, wenn auch der globale Glow-Baustein steht (Stil/Bereich/Ausbreitung/Waerme kommen nur von dort) — die Zeile umfasst darum den globalen Glow mit. Was genau fehlt: (1) eine Camera-Raw-18.6+-Ausgabe (XMP) mit gesetztem globalem Glow, um die crs-Namen und Einheiten von Stil/Bereich/Ausbreitung/Waerme zu kennen; (2) eine Messung je Stil: Graukeil 0..255 und ein heller Punkt bzw. eine Kante auf dunklem Grund, Glow 0/50/100, Ausbreitung und Bereich je 0/50/100, Waerme -100/0/+100 — daraus Profil (Radius in Bildbreite), Schwelle/Gewichtung ueber der Luminanz und Mischart; (3) dieselbe Messung mit Glow nur in einer Maske, um die Verrechnung lokal/global zu sehen. LocalCorrectedDepth und LocalBrightness: keine Fundstelle ausser der Camera-Raw-Ausgabe selbst (Vermutung, nicht belegt: LocalBrightness ist der Regler „Helligkeit" der lokalen Anpassungen aus Prozessversion 2010; LocalCorrectedDepth gehoert zur Tiefenmaske) — es fehlt je eine Adobe-Beschreibung oder eine Messung	fehlt	S
LR14	GPU-Beschleunigung der Entwicklungs-Kette (die Kernel-Infrastruktur steht in Accel)	fehlt	L

#	Lightroom-Funktion	Stand bei dir	Aufw.
LR15	Restposten der Entwicklung (nach LR4/LR5): keine Rauschprofil-Schaetzung je Kamera und keine Wavelet-Baender — gerechnet wird auf sRGB-kodierten Werten; das Schaerfen rechnet seit M1486 mit echtem Gauss-Kern und hat eine Halo-Begrenzung; offen bleibt die Rechnung auf linearen statt sRGB-kodierten Werten und die Kantenschaetzung ueber nur die zentrale Differenz. Bei der Objektivkorrektur fehlen Profildateien (.lcp, Lensfun-XML), sie rechnet rein radial (keine dezentrische Verzeichnung, kein verschobenes Zentrum), kennt nur den lateralen, nicht den axialen Farbfehler und beschneidet nicht automatisch nach dem Entzerren. Ehrlich zur Belegkraft: Lightrooms eigene Schaerfungs- und Entrauschungs-Stufen sind nicht veroeffentlicht und von keinem freien Werkzeug nachgebildet — geprueft sind Wirkung und Richtung, nicht die Uebereinstimmung mit Adobe	fehlt	M
LR18	Restposten von Katalog, Buch und Galerie (nach LR11/LR13): Sammlungen sind die einzige Angabe, die nicht ins Bild geht (sie gehoeren zu einer Auswahl, nicht zu einem Bild) — sie ueberleben deshalb Verschieben und Umbenennen nicht; es gibt keinen Ordner-Baum als eigenes Modell, und RAW wird nur ueber das Sidecar erfasst. Das Fotobuch kennt nur 1/2/4 Bilder je Seite und ein Raster fuers ganze Buch, keine Doppelseiten, keinen Anschnitt, keine Kapitelseiten, und die Unterschrift ist einzeilig; die Galerie hat keine Blaetterung, keinen Lichtkasten, keine Sortierung und zeigt keine Metadaten	fehlt	M
LR19	Restposten des Exports (nach LR16): das Text-Wasserzeichen setzt nur Standard-14, eine Zeile ohne Umbruch, ohne Drehung, Umriss oder Schatten. TIFF wird gelesen, aber nicht zurueckgeschrieben; WebP nimmt nur XMP (EXIF und IPTC werden gemeldet); PNG kennt keinen IPTC-Block. Ehrlich zur Belegkraft: Lightrooms Schaerfungsstufen und die Medien-Zugaben sind nicht veroeffentlicht und von keinem freien Werkzeug nachgebildet — belegt sind Richtung und Ordnung (matt > glaenzend > Bildschirm), nicht die Uebereinstimmung mit Adobe; die Faktoren stehen deshalb als Tabelle, nicht als Formel	fehlt	S

Ehrliche Einordnung: LR1 und LR7 sind die beiden schweren Brocken; LR2/LR3/LR8 sind vergleichsweise geradlinige Bildpunkt-Arbeit auf dem vorhandenen Farb-Unterbau. Ein sinnvoller erster Schnitt waere LR2 + LR3 + LR8 + LR9 — damit gibt es eine echte, nicht-destruktive Entwicklung fuer JPEG/TIFF/PNG, noch ohne RAW.

v4-BSP · Beispiele und Erreichbarkeit

Nichts mehr offen. Die erledigten Punkte samt ihren ehrlich benannten Grenzen stehen in roadmap-erledigt-v4.md.

v4-ROB · Erschoepfung durch begrenzte Eingabe (Befund aus M1451)

Die Fehlerklasse in einem Satz: ein Leser, der jenseits seiner Daten weiterliefert, plus eine Zahl aus dem Strom als Schleifengrenze ergibt

unbegrenzte Arbeit aus begrenzter Eingabe.

M1451 hat das fuer JBIG2 behoben und dabei die Groessenordnung gemessen: 115 Byte Eingabe erzeugten 16 min 28 s Rechenzeit — und am Ende fiel keine Ausnahme, es kam ein Bild heraus. Damit ist das zentrale Projektprinzip als halbe Zusicherung entlarvt: „aus PdfDocument.Load dringt keine Ausnahme" war eingehalten, und der Fall war trotzdem ein Erschoepfungsangriff. „Kommt in endlicher Zeit zurueck" gehoert dazu.

Das Muster ist nicht JBIG2-eigen. Ueberall, wo ein arithmetischer oder Bit-Leser am Datenende normgemaess weiterfuettert (T.88 §E.3.4 und die entsprechenden Stellen in T.800, AV1 und H.265), und ueberall, wo eine Anzahl, Laenge oder Wiederholung aus den Daten gelesen wird, kann dieselbe Luecke stecken. Die vorhandenen Deckel begrenzen jeweils die Zahl, nicht die Erfindung.

Nichts mehr offen (M1506). Alle sechs Punkte stehen mit ihren Messungen in roadmap-erledigt-v4.md.

Ehrlich vorweg: die Deckel (Instanz-, Symbol-, Pixelzahlen) bleiben in jedem Fall noetig — sie sind die Grenze fuer einen Strom, der wirklich Daten nachliefert. ROB ergaenzt sie, ersetzt sie nicht.

v4-EXT · Extern/Toolchain-gebunden (kein fehlender Code)

Aus v3 hierher verschoben — kein fehlender Code, sondern extern/toolchain-/urteilsgebunden.

Punkt	Stand	Was noch fehlt / warum
.wasm-Artefakt (browser-aufrufbar, kpdf_*-Exporte)	zurueckgestellt	JS-Loader + Drift-Waechter (M975) stehen; braucht NativeAOT-LLVM aus dotnet/runtime-Quelle + emscripten-SDK. Auf Nutzerwunsch aufgeschoben.
Native-Shim signieren (Authenticode/codesign)	CI-Schritt	Rezept dist/build-native-shim.sh (M1007) steht; es fehlt nur das Code-Signing-Zertifikat als CI-Secret.
Docker-Hub Untagged-Speicher automatisch freigeben	zurueckgestellt	Tag-Loeschung ist automatisiert (Cleanup Phase A, taeglich). Untagged-Digests belegen ~22 GB; belegt (3 gruene Cleanup-Laeufe lassen sie liegen): Docker Hub hat KEINE oeffentliche REST-API dafuer (Image-Management-UI = session-gebundener Loader). Einzige Automatik = ganzes Repo vor jedem Push loeschen — macht das Repo aber PRIVAT + verliert Beschreibung/Pull-Zaehler. Auf Nutzerwunsch aufgeschoben; bei Bedarf manuell ueber die Web-UI (Image Management) oder Docker-Hubs 6-Monats-GC.