Designing Quoin

A short report on layout, pagination, and deterministic output

SECTIONS	WORDS OF PROSE	ATTRIBUTES
4	509	28
plus a glossary appendix	across 10 justified paragraphs	listed in the reference table

Contents

1. Overview **1**
2. Text and pagination 1
3. Backend and failure modes 2
4. Style attributes 2

Appendix A: Glossary **4**

1. Overview

Quoin starts from a simple observation: most business documents are layouts, not drawings. An invoice is a column of blocks, a letterhead is a row with a flexible gap, and a transcript is a table that happens to be long. The library therefore models documents as trees of components and leaves the coordinate arithmetic to a layout engine.

Every node is immutable. A modifier such as padding or bold returns a new instance, so a fragment can be reused in a dozen places without one caller changing what another sees. This also makes documents safe to build once and render many times, which matters in long-running workers.

Layout follows a measure-then-place model. Each element reports the space it wants given the width on offer, and containers distribute the remainder using grow factors, gaps, and alignment. The vocabulary is deliberately close to flexbox, so anyone who has written modern CSS can predict the result.

2. Text and pagination

Text is shaped with real font metrics. Lines are broken at word boundaries, justified paragraphs stretch the gaps between words, and inline markup such as bold or italic spans is carried through the line breaker without losing its style. Widths come from the embedded font program, never from an estimate.

Pagination is a first-class concern rather than an afterthought. Blocks can ask to be kept together, a heading can insist on staying with the paragraph that follows it, and tables repeat their header on every page they cross. When nothing else works, the engine splits a block at a line boundary and keeps going.

Output is deterministic. Given the same input and a fixed creation date, two runs produce identical bytes. That property makes golden-file tests practical: a visual regression becomes a byte comparison, and a diff in the PDF points at a diff in the code.

3. Backend and failure modes

The backend is a thin seam. The layout engine produces a display list of rectangles, lines, text runs, and links in plain points with a top-left origin, and one adapter turns that list into PDF objects. Nothing above the adapter knows how a PDF is encoded, so the engine can be tested without parsing a single file.

Failure is explicit. If a block cannot fit on a page even after every permitted split, the engine throws by default and names the offending node. Callers who prefer degraded output can opt into clipping, which keeps the rest of the document flowing and reports every clipped node through a warning callback; silent corruption is never the default.

Styling is data. Colors resolve from a named palette, sizes accept points, millimeters, or percentages, and a theme supplies defaults for family, size, weight, and line height. Because every style property can also be set from a string attribute, templates written in markup and code written in PHP describe the same documents.

The result is a library that stays out of the way for simple documents and still holds together for hard ones: a forty-page ledger, a certificate with a decorative background, or a report whose tables must not be orphaned from their captions.

Rule of thumb: **measure first**, then *place*, and only then paint.

4. Style attributes

Attribute	Example	Effect
padding	12pt 6mm	Space inside the border; one to four values, like CSS.
margin	0 0 6pt	Space outside the border; one to four values.
gap	8pt	Space between the children of a row or column.
grow	1	Share of the free space a child takes along the main axis.
shrink	0	Share of the overflow a child gives back along the main axis.
basis	40%	Starting main size of a flex child before grow and shrink.
width	60mm	Border-box width: a length, a percentage, or auto.
height	120pt	Border-box height; a fixed height makes the box monolithic.

Attribute	Example	Effect
min-width / max-width	20mm	Clamps applied after the width is resolved.
align-items	center	Cross-axis placement of the children.
justify	space-between	Main-axis distribution of the free space.
background	slate-50	Fill color: a palette name or a hex value.
border-width	1pt	Border width; one to four values.
border-color	slate-200	Border color: a palette name or a hex value.
radius	8pt	Corner radius of the background and border.
opacity	0.5	Opacity of the box and everything inside it.
font	Inter	Font family, resolved through the font registry.
font-size	10pt	Font size.
weight	700	Font weight, such as 400 or 700.
color	slate-900	Text color: a palette name or a hex value.
line-height	1.4	Line height as a multiple of the font size.
letter-spacing	0.5pt	Extra space after every character.
text-align	justify	Left, center, right or justify.
underline / strike	true	Text decorations drawn under or through the run.
keep-together	true	Moves the whole block to the next page rather than splitting it.
keep-with-next	true	Keeps a block on the same page as the block that follows it.
break-before / break-after	true	Forces a page break before or after the block.
min-lines	2	Fewest lines of a paragraph left on either side of a page break.

Appendix A: Glossary

Display list: The ordered drawing operations produced by layout.

Fragment: The part of a node that fits on one page.

Track: A column width definition in a table.