TERRAPDF DEVELOPER GUIDE

Introduction

This guide provides a comprehensive overview of TerraPDF, a pure-C# document generation library with zero runtime dependencies. TerraPDF enables developers to create professional-quality PDF documents programmatically using a fluent, composable API. From simple text blocks to complex multi-page tables with repeating headers, TerraPDF handles all aspects of document layout, styling, and content pagination.

Getting Started

To begin using TerraPDF, add the NuGet package to your project and start building documents with the fluent API. The library supports custom page sizes, margins, headers, footers, and a rich set of text formatting options including bold, italic, underline, strikethrough, and per-span styling. Page numbers can be inserted as dynamic placeholders.

GETTING STARTED

Installation

Install TerraPDF via NuGet: dotnet add package TerraPDF. The library targets .NET 8.0 and .NET 9.0, with no external dependencies. All PDF generation is done using pure managed code, ensuring compatibility across all platforms supported by .NET.

Quick Start

Create your first PDF in minutes with the fluent builder pattern. Start with Document.Create(), define one or more pages, and add content to the header, content, and footer slots. The library automatically handles multi-page pagination, table splitting, and page numbering.

Configuration

Every PageDescriptor offers extensive configuration: page size (A0-A6, Letter, Legal, or custom), margins (uniform or per-edge), background colour, and a default text style that cascades to all child elements. Use the HeaderOnFirstPageOnly() method to restrict header rendering to the first page of a multi-page section.

CORE FEATURES – TEXT & TYPOGRAPHY

Text & Typography

TerraPDF provides comprehensive typographic controls: font size, colour (hex or Material Design palette), bold, italic, underline, strikethrough, and line-height spacing. The TextBlock element tokenises input into spans, allowing mixed formatting within a single paragraph. Alignment options include left, centre, right, and justified.

Layout Containers

Three core layout containers enable flexible page composition:
• Column — vertically stacks items with configurable spacing; auto-paginates.
• Row — arranges children horizontally with relative/constant sizing.
• Table — grid layout with header rows that repeat on continuation pages.
All containers support decorators: Padding, Margin, Background, Border, and Alignment.

Images & Hyperlinks

Embed PNG (RGB) and JPEG images directly. TerraPDF decodes PNGs and recompresses them with FlateDecode; JPEGs are embedded verbatim. Hyperlink elements create URI annotations — clickable areas that open external URLs in the viewer.

CORE FEATURES – TABLES & DECORATORS

Tables

Tables are the most powerful layout primitive in TerraPDF. Define columns using relative widths (proportional to available space) or fixed point sizes. Header rows are automatically repeated on every page when a table spans multiple pages. Cells support column-span and row-span for merged layouts. Rows are rendered with consistent heights across page breaks, ensuring predictable pagination.

Decorators

Every container and element can be wrapped with decorators that modify its box model. Padding adds inner spacing; Margin adds outer spacing; Background fills the content area; Border draws edges; Alignment changes the positioning context. Decorators chain freely and are applied in a well-defined order: Margin ? Padding ? Background ? Border.

ADVANCED TOPICS

Multi-Page Documents

TerraPDF uses a two-pass rendering strategy: the first pass measures all content to determine the total page count (enabling accurate page-number placeholders), and the second pass emits the final PDF objects. Columns automatically split across pages when content overflows; explicit PageBreak() elements force a new page at any point.

Custom Styling

The TextStyle value object is immutable — every mutating method returns a new instance. This copy-on-write pattern ensures that styles are safely shared across elements without accidental mutation. Use the DefaultTextStyle() method on PageDescriptor to establish a base style, then override selectively on individual elements.

Performance Tips

TerraPDF is designed for high-throughput scenarios: PDFs are generated entirely in-memory with efficient string builders and binary stream compression. Avoid repeatedly creating identical styled TextBlocks — reuse IComponent implementations when the same content block appears on multiple pages or in multiple places.

APPENDIX

API Reference

The TerraPDF API surface is intentionally small and composable. Key entry points:
• Document.Create() — starts a new document builder.
• IDocumentContainer.Page() — configures a page via PageDescriptor.
• IContainer methods: Text(), Column(), Row(), Table(), Image(), Hyperlink().
• TextStyle modifiers: Bold(), Italic(), Underline(), FontSize(), FontColor().
All public methods validate arguments and throw appropriate exceptions for invalid input.

Sample Code

```
var pdf = Document.Create(doc =>
{
doc.Page(page =>
{
page.Size(PageSize.A4);
page.Content().Text("Hello");
});
}).PublishPdf("output.pdf");
```

Migration Guide

Upgrading from TerraPDF 1.x to 2.0? Key breaking changes include the new descriptor pattern (PageDescriptor, TextDescriptor, SpanDescriptor), removal of the legacy fluent API methods in favour of explicit configuration objects, and the switch to immutable TextStyle. See the online migration guide for a detailed diff and automated upgrade scripts.

Thank you for choosing TerraPDF!