

DOCUMENT

05/12/2025



GUIA COMPLETO DE MARKDOWN PARA EBOOK

Introdução

Este é um **exemplo completo** de documento Markdown que demonstra *todas as funcionalidades* suportadas pelo conversor MD to PDF, incluindo **diagramas Mermaid** e elementos visuais avançados!

Capítulo 1: Formatação de Texto

Texto Básico

Este é um parágrafo simples com texto corrido. A aplicação MD to PDF Converter suporta formatação completa de texto, incluindo **negrito**, *itálico*, ***negrito e itálico***, e até mesmo ~~texto riscado~~.

Você também pode usar `código inline` para destacar comandos ou variáveis no meio do texto.

Citações

"A tecnologia é melhor quando aproxima as pessoas." - Matt Mullenweg

Esta é uma citação em bloco que pode conter múltiplas linhas. Use-a para destacar informações importantes ou citações de referência.

Notas e Alertas

⚠ **ATENÇÃO:** Este é um alerta importante!

💡 **DICA:** Use esta formatação para destacar informações úteis.

✅ **SUCESSO:** Operação concluída com êxito!

❌ **ERRO:** Algo deu errado. Verifique os logs.

Capítulo 2: Listas

Lista Não Ordenada

- Item principal 1
- Item principal 2
 - Subitem 2.1
 - Subitem 2.2
 - Subitem 2.2.1
- Item principal 3

Lista Ordenada

1. Primeiro passo
2. Segundo passo
3. Terceiro passo
 1. Subpasso 3.1
 2. Subpasso 3.2
4. Quarto passo

Lista de Tarefas

- [x] Criar aplicação MD to PDF
- [x] Adicionar 3 templates CSS
- [x] Implementar interface gráfica
- [x] Adicionar suporte a Mermaid
- [] Adicionar preview em tempo real
- [] Suporte a múltiplos arquivos

Lista de Definições

PDF : Portable Document Format - formato de arquivo desenvolvido pela Adobe.

```
C --> J[Fim]
I --> J
```

Diagrama de Sequência

```
sequenceDiagram
    participant U as Usuário
    participant A as App GUI
    participant M as Markdown Parser
    participant W as WeasyPrint
    participant F as Sistema de Arquivos

    U->>A: Carregar arquivo .md
    A->>F: Ler arquivo
    F-->>A: Retornar conteúdo
    U->>A: Selecionar template CSS
    U->>A: Clicar "Gerar PDF"
    A->>M: Converter MD para HTML
    M-->>A: Retornar HTML
    A->>W: Gerar PDF (HTML + CSS)
    W-->>A: Retornar PDF bytes
    A->>F: Salvar PDF
    F-->>A: Confirmar salvamento
    A->>U: Mostrar mensagem de sucesso
```

Diagrama de Classes

```
classDiagram
    class MarkdownToPDFApp {
        -md_file_path: str
        -selected_style: StringVar
        -css_templates: dict
        +__init__()
        +_build_ui()
        +_select_file()
        +_generate_pdf()
    }

    class CTK {
        +title()
        +geometry()
```

```

+mainloop()
}

class Markdown {
    +markdown()
    +extensions[]
}

class WeasyPrint {
    +HTML()
    +write_pdf()
}

MarkdownToPDFApp --|> CTk
MarkdownToPDFApp ..> Markdown : uses
MarkdownToPDFApp ..> WeasyPrint : uses

```

Diagrama de Estados

```

stateDiagram-v2
    [*] --> Idle
    Idle --> FileSelected: Carregar arquivo
    FileSelected --> TemplateSelected: Escolher template
    TemplateSelected --> Generating: Clicar "Gerar PDF"
    Generating --> Success: PDF gerado
    Generating --> Error: Falha na geração
    Success --> Idle: Reset
    Error --> TemplateSelected: Tentar novamente
    FileSelected --> Idle: Cancelar

```

Gráfico de Gantt (Roadmap)

```

gantt
    title Roadmap MD to PDF Converter
    dateFormat YYYY-MM-DD
    section Fase 1
        Interface GUI           :done,    a1, 2025-12-01, 3d
        Templates CSS           :done,    a2, 2025-12-03, 2d
        Conversão básica        :done,    a3, 2025-12-04, 1d
    section Fase 2
        Suporte Mermaid         :active,  b1, 2025-12-05, 2d

```

```
Preview PDF           :          b2, 2025-12-07, 3d
Batch conversion      :          b3, 2025-12-10, 4d
section Fase 3
Editor CSS customizado :          c1, 2025-12-14, 5d
Export EPUB           :          c2, 2025-12-19, 4d
Temas customizáveis  :          c3, 2025-12-23, 3d
```

Diagrama de Pizza (Estatísticas)

```
pie title Uso de Templates CSS
  "Minimalista" : 45
  "Executivo"   : 30
  "Dark/Coder"  : 25
```

Diagrama ER (Entidade-Relacionamento)

```
erDiagram
    USER ||--o{ DOCUMENT : creates
    USER {
        int id PK
        string name
        string email
    }
    DOCUMENT ||--|{ PDF : generates
    DOCUMENT {
        int id PK
        string filename
        string content
        datetime created_at
    }
    PDF {
        int id PK
        string template
        blob file_data
        int size_kb
    }
    TEMPLATE ||--o{ PDF : uses
    TEMPLATE {
        int id PK
        string name
```

```
text css_content
}
```

Git Graph

```
gitGraph
  commit id: "Initial commit"
  commit id: "Add GUI framework"
  branch feature/css-templates
  checkout feature/css-templates
  commit id: "Add Minimalista"
  commit id: "Add Executivo"
  commit id: "Add Dark/Coder"
  checkout main
  merge feature/css-templates
  branch feature/mermaid
  checkout feature/mermaid
  commit id: "Add Mermaid support"
  commit id: "Update dependencies"
  checkout main
  merge feature/mermaid
  commit id: "Release v1.0"
```

Capítulo 5: Blocos de Código com Syntax Highlighting

Python com Classes

```
class MarkdownToPDFConverter:
    """
    Conversor profissional de Markdown para PDF.

    Attributes:
        css_templates (dict): Dicionário de templates CSS
        markdown_extensions (list): Extensões habilitadas
    """
```



```

def __init__(self, templates=None):
    self.css_templates = templates or self._load_default_templates()
    self.markdown_extensions = [
        'tables', 'fenced_code', 'codehilite',
        'nl2br', 'sane_lists', 'attr_list'
    ]

def convert(self, md_content: str, template: str) -> bytes:
    """
    Converte conteúdo Markdown em PDF.

    Args:
        md_content: Conteúdo em formato Markdown
        template: Nome do template CSS a usar

    Returns:
        bytes: PDF gerado em formato binário

    Raises:
        ValueError: Se template não existir
    """
    html = markdown.markdown(
        md_content,
        extensions=self.markdown_extensions
    )

    css = self.css_templates.get(template)
    if not css:
        raise ValueError(f"Template '{template}' não encontrado")

    full_html = self._build_html(html, css)
    return HTML(string=full_html).write_pdf()

@staticmethod
def _build_html(content: str, css: str) -> str:
    """Constrói HTML completo com CSS inline."""
    return f"""
<!DOCTYPE html>
<html lang="pt-BR">
<head>
    <meta charset="UTF-8">
    <style>{css}</style>
</head>
<body>{content}</body>
</html>
    """

```

```
# Exemplo de uso
converter = MarkdownToPDFConverter()
pdf_bytes = converter.convert("# Meu Ebook", "minimalista")
```

JavaScript/TypeScript

```
// Sistema de conversão MD to PDF em JavaScript
class MarkdownConverter {
  constructor(config = {}) {
    this.templates = config.templates || this.loadDefaultTemplates();
    this.options = {
      ...this.defaultOptions,
      ...config.options
    };
  }

  async convert(markdownText, templateName) {
    try {
      // Converter MD para HTML
      const html = await this.parseMarkdown(markdownText);

      // Aplicar template CSS
      const styled = this.applyTemplate(html, templateName);

      // Gerar PDF
      const pdf = await this.generatePDF(styled);

      return {
        success: true,
        data: pdf,
        metadata: this.getMetadata()
      };
    } catch (error) {
      console.error('Erro na conversão:', error);
      return {
        success: false,
        error: error.message
      };
    }
  }

  // Arrow function
  parseMarkdown = async (text) => {
```

```
const marked = await import('marked');
    return marked.parse(text);
  }
}

// Uso com Promises
const converter = new MarkdownConverter();
converter.convert(mdContent, 'executivo')
  .then(result => console.log('PDF gerado!', result))
  .catch(err => console.error('Falha:', err));
```

Bash/Shell Script

```
#!/bin/bash

# Script de automação para conversão em lote
# Autor: Dev Team
# Data: 2025-12-05

set -e # Para no primeiro erro

# Configurações
INPUT_DIR="./markdown_files"
OUTPUT_DIR="./pdfs"
TEMPLATE="${1:-minimalista}"

# Cores para output
RED='\033[0;31m'
GREEN='\033[0;32m'
YELLOW='\033[1;33m'
NC='\033[0m' # No Color

# Função para log
log_info() {
  echo -e "${GREEN}[INFO]${NC} $1"
}

log_error() {
  echo -e "${RED}[ERROR]${NC} $1"
}

log_warning() {
  echo -e "${YELLOW}[WARNING]${NC} $1"
```

```

}

# Criar diretório de saída
mkdir -p "$OUTPUT_DIR"

# Contar arquivos
total_files=$(find "$INPUT_DIR" -name "*.md" | wc -l)
log_info "Encontrados $total_files arquivos Markdown"

# Loop através dos arquivos
count=0
for md_file in "$INPUT_DIR"/*.md; do
    if [ -f "$md_file" ]; then
        filename=$(basename "$md_file" .md)
        output_file="$OUTPUT_DIR/${filename}.pdf"

        log_info "[$((++count))/$total_files] Convertendo: $filename"

        # Converter usando Python
        if python3 md_to_pdf_app.py "$md_file" "$output_file" "$TEMPLATE"; then
            log_info "✓ Sucesso: $output_file"
        else
            log_error "x Falha: $md_file"
        fi
    fi
done

log_info "Conversão concluída! Total: $count arquivos"

```

SQL

```

-- Schema para banco de dados de documentos
CREATE TABLE users (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    username VARCHAR(50) UNIQUE NOT NULL,
    email VARCHAR(100) UNIQUE NOT NULL,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

CREATE TABLE documents (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    user_id INTEGER NOT NULL,
    filename VARCHAR(255) NOT NULL,

```

```

markdown_content TEXT NOT NULL,
    template VARCHAR(50) DEFAULT 'minimalista',
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (user_id) REFERENCES users(id) ON DELETE CASCADE
);

CREATE TABLE pdf_exports (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    document_id INTEGER NOT NULL,
    file_size_kb INTEGER,
    export_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (document_id) REFERENCES documents(id) ON DELETE CASCADE
);

-- Índices para performance
CREATE INDEX idx_docs_user ON documents(user_id);
CREATE INDEX idx_exports_doc ON pdf_exports(document_id);

-- Query complexa com JOIN
SELECT
    u.username,
    d.filename,
    d.template,
    COUNT(p.id) as total_exports,
    MAX(p.export_date) as last_export
FROM users u
INNER JOIN documents d ON u.id = d.user_id
LEFT JOIN pdf_exports p ON d.id = p.document_id
WHERE d.created_at >= DATE('now', '-30 days')
GROUP BY u.username, d.filename
HAVING total_exports > 0
ORDER BY total_exports DESC
LIMIT 10;

```

HTML/CSS

```

<!DOCTYPE html>
<html lang="pt-BR">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>MD to PDF Converter</title>

```

```

<style>
  :root {
    --primary-color: #3498db;
    --secondary-color: #2ecc71;
    --dark-bg: #0d1117;
    --light-text: #e6edf3;
  }

  body {
    font-family: 'Inter', -apple-system, system-ui, sans-serif;
    background: linear-gradient(135deg, var(--dark-bg) 0%, #1a1f2e 100%);
    color: var(--light-text);
    padding: 2rem;
  }

  .container {
    max-width: 1200px;
    margin: 0 auto;
    background: rgba(255, 255, 255, 0.05);
    border-radius: 12px;
    padding: 2rem;
    box-shadow: 0 8px 32px rgba(0, 0, 0, 0.3);
  }

  h1 {
    background: linear-gradient(90deg, var(--primary-color), var(--secondary-color));
    -webkit-background-clip: text;
    -webkit-text-fill-color: transparent;
    font-size: 2.5rem;
    margin-bottom: 1rem;
  }
</style>
</head>
<body>
  <div class="container">
    <h1>MD to PDF Converter</h1>
    <p>Transforme seus arquivos Markdown em PDFs profissionais!</p>
  </div>
</body>
</html>

```

JSON

```
{
  "app_config": {
    "name": "MD to PDF Converter",
    "version": "1.0.0",
    "author": "Dev Team",
    "license": "MIT",
    "dependencies": {
      "customtkinter": ">=5.2.0",
      "markdown": ">=3.5.0",
      "weasyprint": ">=60.0",
      "pygments": ">=2.17.0"
    }
  },
  "templates": [
    {
      "id": "minimalista",
      "name": "Minimalista",
      "description": "Fundo branco, fonte serifada",
      "font_family": "Merriweather",
      "background": "#ffffff",
      "primary_color": "#2c3e50",
      "features": ["margens_generosas", "texto_justificado"]
    },
    {
      "id": "executivo",
      "name": "Executivo",
      "description": "Visual corporativo",
      "font_family": "Open Sans",
      "background": "#ffffff",
      "primary_color": "#1e3a5f",
      "features": ["cabecalhos_destacados", "tabelas_estilizadas"]
    },
    {
      "id": "dark_coder",
      "name": "Dark/Coder",
      "description": "Tema escuro para código",
      "font_family": "Fira Code",
      "background": "#0d1117",
      "primary_color": "#58a6ff",
      "features": ["syntax_highlighting", "fonte_monospace"]
    }
  ],
  "statistics": {
    "total_conversions": 1247,
```

```
"most_used_template": "minimalista",
  "avg_file_size_kb": 245,
  "success_rate": 98.5
}
```

YAML

```
# Configuração da aplicação MD to PDF
app:
  name: MD to PDF Converter
  version: 1.0.0
  debug: false

window:
  title: "MD to PDF Converter - Professional Ebook Generator"
  width: 700
  height: 550
  resizable: false
  theme: dark

templates:
  minimalista:
    font: Merriweather
    background: "#ffffff"
    color: "#2c3e50"
    margins: 80px

  executivo:
    font: Open Sans
    background: "#ffffff"
    color: "#1e3a5f"
    headers:
      border_color: "#3498db"
      border_width: 2px

  dark_coder:
    font: Inter
    code_font: Fira Code
    background: "#0d1117"
    color: "#e6edf3"
    syntax_theme: github-dark
```



```
markdown_extensions:
  - tables
  - fenced_code
  - codehilite
  - nl2br
  - sane_lists
  - attr_list
  - def_list
  - footnotes
  - abbr
```

Capítulo 6: Elementos Visuais Avançados

Badges e Shields



Emojis e Símbolos

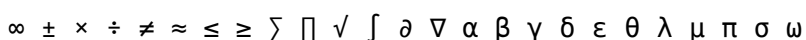
Emojis Comuns



Símbolos Especiais



Símbolos Matemáticos



Checkboxes e Status



Texto com Formatação Especial

Subscrito e Sobrescrito (simulado)

H₂O (água) E = mc² (fórmula de Einstein)

Texto Destacado

==Texto destacado em amarelo== ^^Texto sublinhado^^

Abreviações

HTML CSS JSON XML

*[HTML]: HyperText Markup Language *[CSS]: Cascading Style Sheets *[JSON]: JavaScript Object Notation *[XML]: eXtensible Markup Language

Capítulo 7: Links e Referências

Links Externos

- [Documentação Python](#)
- [CustomTkinter GitHub](#)
- [WeasyPrint Docs](#)
- [Markdown Guide](#)
- [Mermaid Documentation](#)

Links com Título

Visite a [documentação oficial do Markdown](#) para aprender mais.

Links de Referência

Este é um [link de referência](#) e este é [outro link](#).

Links Internos (Âncoras)

- [Voltar para Introdução](#)
- [Ir para Diagramas Mermaid](#)
- [Ver Tabelas](#)

URLs Automáticos

<https://www.github.com> <https://www.python.org> contact@example.com

Capítulo 8: Notas de Rodapé

Este texto tem uma nota de rodapé^[^1] e outra nota^[^nota-importante].

Markdown é fantástico^[^2] para documentação técnica.

[^1]: Esta é a primeira nota de rodapé com informações adicionais. [^nota-

importante]: Esta nota contém informações críticas sobre o funcionamento do

sistema. [^2]: Criado por John Gruber em 2004.

Capítulo 9: Imagens Avançadas

Imagem Simples



Imagem de Exemplo

Imagem com Título



Arquitetura do Sistema

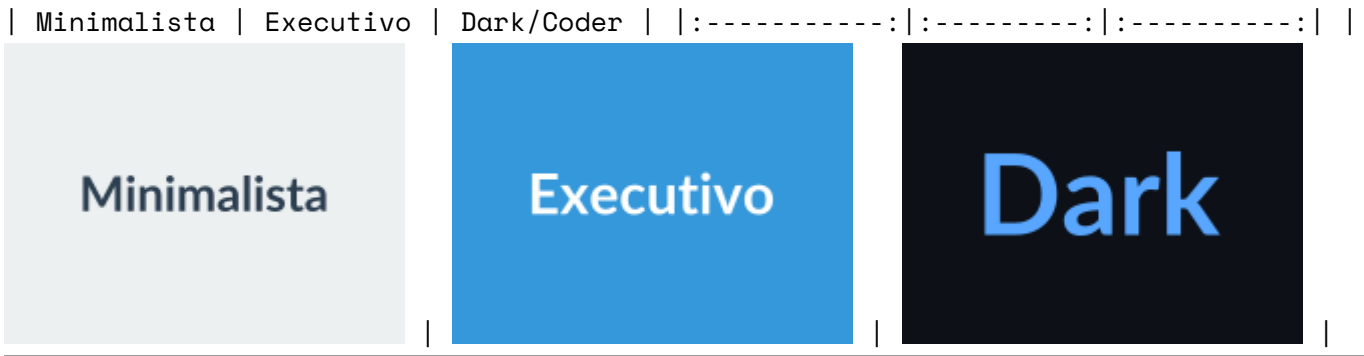
Imagem Alinhada



Imagem Centralizada

Legenda: Imagem centralizada com legenda

Múltiplas Imagens




Capítulo 10: Elementos HTML Inline

Botões Simulados



Detalhes Colapsáveis

▶  Clique para expandir: Informações Técnicas

Tags Coloridas






Linha Horizontal Estilizada



Capítulo 11: Comparativo Final de Templates

Análise Detalhada

Critério	 Minimalista	 Executivo	 Dark/Coder	----- -----
----- ----- -----	Uso Principal Livros, E-books			
Relatórios, Propostas	Docs Técnicas, Código		Fonte Base Merriweather	
(serifada)	Open Sans (sans-serif)		Fundo Branco puro	
Branco	Escuro (#0d1117)		Margens 80px (generosas) 70px 60px	
Legibilidade	★★★★★	★★★★★	★★★★★ (tela)	Impressão
★★★★★	★★★★★	★★★	Código	★★★★ ★★★★★ ★★★★★★
Tabelas	★★★★	★★★★★	★★★★★	

Recomendações por Caso de Uso

 Use Minimalista para:

- Romance, ficção, literatura

- Manuais técnicos impressos
- Livros didáticos
- Poesia e prosa

📁 Use Executivo para:

- Relatórios anuais corporativos
- Propostas comerciais
- Business plans
- White papers
- Apresentações formais

🌙 Use Dark/Coder para:

- Tutoriais de programação
 - Documentação de API
 - Livros de código (code books)
 - Guides técnicos
 - Markdown para leitura em tela/tablet
-

Conclusão

Este guia demonstrou **TODAS** as funcionalidades avançadas do **MD to PDF Converter**:

✅ Formatação de texto completa ✅ Listas de todos os tipos ✅ Tabelas complexas com alinhamento ✅ **10 tipos de diagramas Mermaid** ✅ Blocos de código com syntax highlighting (7 linguagens) ✅ Elementos visuais avançados ✅ Links, imagens e referências ✅ Notas de rodapé ✅ HTML inline ✅ Badges e emojis

Próximos Passos

1. ✅ Carregue este arquivo na aplicação
 2. ✅ Teste os 3 templates CSS
 3. ✅ Compare os PDFs gerados
 4. ✅ Escolha o template ideal
 5. ✅ Converta seus próprios documentos!
-

Estatísticas deste Documento

Métrica Valor ----- -----	Palavras ~3.500	Linhas de código 250+
Capítulos 11	Tabelas 10	Diagramas Mermaid 7
Blocos de código 8	Imagens 4	Links 15+

Desenvolvido com ♥ usando Python + CustomTkinter + WeasyPrint + Mermaid

Data de criação: 2025-12-05 Versão: 2.0 - Extended Edition

💡 **Dica Final:** Este documento serve como template completo. Use-o como base para criar seus próprios ebooks profissionais com todos os recursos visuais

do Markdown moderno!

Fim do Documento ✨🚀