Documentação Técnica: Processo de Criação de Tabela

Objetivo: Esta documentação descreve, passo a passo, o fluxo de criação de uma nova tabela dentro da aplicação `GridFlow`. O objetivo é fornecer um guia detalhado para desenvolvedores, cobrindo desde a seleção da planilha até a geração e armazenamento dos arquivos finais (XML, CSV, PDF).

---

1. Visão Geral do Fluxo

O processo de criação de uma tabela é dividido em etapas sequenciais, guiadas por interfaces gráficas (Views) e orquestradas por Controllers e Services. O fluxo principal é: `Seleção da Planilha` → `Configuração Inicial (Dimensões e Cabeçalho)` → `Edição dos Dados` → `Decisão sobre Relatório (Opcional)` → `Geração e Armazenamento dos Arquivos (XML, CSV, PDF)`.

O objetivo principal é permitir que o usuário defina a estrutura (dimensões, cabeçalho) e o conteúdo de uma nova tabela, e que o sistema persista esses dados em múltiplos formatos, organizando-os em uma estrutura de diretórios específica.

---

2. Pré-requisitos

- O diretório `.../executavel/Planilhas/` deve existir e conter pelo menos uma subpasta representando uma planilha válida (ex: `.../executavel/Planilhas/FinanceiroMaio2025/`).

---

3. Etapa 1: Seleção da Planilha (`TViewSelecionarPlanilhaParaTabela`)

- **Interface:** Uma view modal contendo um `TListBox` ou `TComboBox` listando as planilhas disponíveis.
- **Lógica de Listagem:** O controller (ou a própria view) varre o diretório `.../executavel/Planilhas/` e lista os nomes das subpastas (que representam as planilhas).
- **Ação "Avançar":** Ao clicar em "Avançar" com uma planilha selecionada, a view instancia e exibe `TViewConfigurarTabela`, passando o nome da planilha selecionada como parâmetro. A view atual é então fechada.

---

4. Etapa 2: Configuração Inicial da Tabela (`TViewConfigurarTabela`)

- **Interface:** Uma view contendo:
  - Dois `TEdit` para entrada do número de **Linhas** e **Colunas** (com validação para números inteiros positivos).
  - Um `TRadioGroup` com as opções "Cabeçalho na Primeira **Linha**" e "Cabeçalho na Primeira **Coluna**".
  - Botões "Avançar", "Voltar", "Cancelar".
- **DTO de Configuração:** Um `TConfiguracaoTabelaDTO` é criado para transportar os dados desta etapa.
  pascal

  ```
  FNumLinhas: Integer;
  FNumColunas: Integer;
  FTipoCabecalho: TTipoCabecalho;
  FPlanilhaNome: string; // Nome da planilha pai
  end;
  ```

- **Importância da Escolha do Cabeçalho:**
  - Esta escolha define como os dados serão interpretados semanticamente.
  - O CSS fornecido (`estilo.css`) e o serviço de geração de PDF (`FPDFService`) estão atualmente otimizados para o cenário onde o cabeçalho está na **primeira linha** (`Linha[numero="1"] Celula`).
  - A implementação para o cabeçalho na primeira coluna é uma melhoria futura. Por enquanto, o sistema pode forçar `tcLinha` ou tratar `tcColuna` como `tcLinha` para evitar quebras.
- **Ações:**
  - **"Avançar":** Valida as entradas. Cria um `TConfiguracaoTabelaDTO`, preenche-o com os dados da interface e o nome da planilha. Instancia `TViewEditorTabela` passando este DTO e exibe a nova view. Fecha a view atual.
  - **"Voltar":** Fecha a view atual e reabre `TViewSelecionarPlanilhaParaTabela`.
  - **"Cancelar":** Fecha toda a cadeia de views, cancelando o processo de criação.

---

5. Etapa 3: Edição dos Dados da Tabela (`TViewEditorTabela` - Modo Criação)

- **Interface:** Reutiliza a view `TViewEditorTabela` existente, mas em um modo de "criação" (não edição de um arquivo existente).

- **Configuração do `TClientDataSet`:**
    - O método `ConfigurarClientDataSet` é modificado para aceitar um `TConfiguracaoTabelaDTO`.
    - Ele cria `FNumColunas` campos (`TFieldDef`) no `ClientDataSet`.
    - Deve inserir `FNumLinhas` registros vazios.
    - O `DBGrid` será populado automaticamente com esta estrutura.
- **Interação do Usuário:** O usuário interage com o `DBGrid` para preencher os dados da tabela.
- **Título da Tabela:** Um `TEdit` (`EditarTituloTabela`) permite ao usuário definir o título da nova tabela.
- **Ações:**
    - **"Salvar":** Coleta o título do `TEdit`. Chama o método `ExecutarSalvarComConfirmacao`, que por sua vez invoca o `TCriadorTabelaController.CriarNovaTabela`, passando o DTO da tabela (com o título) e o `TClientDataSet` preenchido.
    - **"Cancelar":** Fecha a view, cancelando a criação.

---

6. Etapa 4: Decisão sobre Relatório (Opcional)

- **Localização:** Esta decisão é implementada como parte do processo dentro do `TCriadorTabelaController.CriarNovaTabela`, após o usuário clicar em "Salvar" na `TViewEditorTabela`.
- **Interface:** Um diálogo modal pergunta "Deseja aplicar um relatório a esta tabela?". Se "Sim", exibe um `TOpenDialog` ou `TListBox` para o usuário selecionar um arquivo `.csv` de relatório do diretório `.../executavel/Relatorios/`.
- **Conceito de Relatório:** Um relatório é um conjunto de instruções (definidas em um `.csv`) que reorganizam ou formatam os dados da tabela *antes* de sua persistência final. Ex: reordenar colunas, ocultar linhas, aplicar máscaras.

---

7. Etapa 5: Geração e Armazenamento dos Arquivos (`TCriadorTabelaController.CriarNovaTabela`)

- **Nomeação dos Arquivos:**
    1. `TituloFormatado := CapitalizarPrimeiraLetra(ATabelaDTO.Titulo);`
    2. `PlanilhaFormatada := CapitalizarPrimeiraLetra(AConfiguracao.PlanilhaNome);`
    3. `NomeBaseArquivo := TituloFormatado + PlanilhaFormatada;` (Ex: `Dia1FinanceiroMaio2025`)
- **Criação de Diretórios:**

1. **Caso "Não" (Relatório):**
    - Diretório Base: `.../executavel/Planilhas/<NomeDaPlanilha>/Tabelas/`
    - Diretório Específico: `.../executavel/Planilhas/<NomeDaPlanilha>/Tabelas/<TituloFormatado>/`
2. **Caso "Sim" (Relatório):**
    - Diretório Base: `.../executavel/TabelaRelatorio/`
    - Diretório Específico: `.../executavel/TabelaRelatorio/<TituloFormatado><PlanilhaFormatada>_<TituloDoRelatorio>/`

- **Geração dos Arquivos:**
    1. **XML (`FXMLService.GravarXML`):**
        - Gera o arquivo `.xml` no diretório específico.
        - Inclui os atributos `numero` nas tags `<Linha>` e `<Celula>`.
        - Usa o `TTipoCabecalho` da configuração para estruturar semanticamente o XML (atualmente focado em `tcLinha`).
    2. **CSV (`FCSVService.GravarCSV`):**
        - Gera o arquivo `.csv` no diretório específico.
        - Formato padrão: primeira linha com nomes de colunas, dados separados por vírgulas.
    3. **PDF (`FPDFService.GerarPDF`):**
        - Gera o arquivo `.pdf` no diretório específico.
        - Utiliza o arquivo XML gerado e o arquivo CSS `.../executavel/estilo.css` como entrada.
        - *(Implementação real de `FPDFService` é necessária, ex: usando FastReport).*
- **Mensagem ao Usuário:** Após a geração bem-sucedida, exibir uma mensagem informando os caminhos completos dos três arquivos gerados.

---

8. Detalhamento da Geração de PDF

- **Processo:** O serviço `FPDFService` deve:
    1. Ler o arquivo XML gerado na etapa anterior.
    2. Aplicar as regras de estilo definidas no `estilo.css`.
    3. Renderizar o conteúdo formatado em um arquivo PDF.
- **Tecnologia:** Deve ser implementado usando uma biblioteca de terceiros como **FastReport**, **QuickReport** ou **FortesReport**. A implementação atual é apenas um stub.
- **Entrada:** Caminho do arquivo XML e caminho do arquivo `estilo.css`.
- **Saída:** Caminho do arquivo PDF de destino.

---

9. Detalhamento da Aplicação de Relatórios

- **Formato do Arquivo de Relatório (.csv):**
  - **Proposta 1 (Reordenação Simples):** Uma única linha contendo os índices das colunas na ordem desejada.
    - Exemplo: `1,4,2,3,5` (move a coluna 4 para a posição 2).
  - **Proposta 2 (Ações Estruturadas):** Um arquivo CSV com cabeçalho definindo o tipo de ação.
    - Exemplo:

    - 1
      2
      3
      4
      Tipo,Acao,Origem,Destino
      Coluna,Mover,4,2
      Linha,Ocultar,5,
      Celula,Formatar,3,2,"R$ #,##0.00"

- **Leitura do Relatório:** O `TCriadorTabelaController` (ou um `TRelatorioService` dedicado) deve ler o arquivo `.csv` selecionado e interpretar suas instruções.
- **Transformação dos Dados:** Antes de chamar `FXMLService.GravarXML` e `FCSVService.GravarCSV`, o controller deve modificar o `TClientDataSet` (ou criar uma cópia) conforme as instruções do relatório. Por exemplo, reordenar as colunas com base no `.csv`.
- **Impacto no Armazenamento:** Como mencionado, os arquivos são salvos em `.../executavel/TabelaRelatorio/...` para diferenciar tabelas com relatórios aplicados.

---

10. Fluxo de Dados entre Componentes

1. `TViewSelecionarPlanilhaParaTabela` coleta o nome da planilha e passa para `TViewConfigurarTabela`.
2. `TViewConfigurarTabela` coleta as dimensões e tipo de cabeçalho, cria um `TConfiguracaoTabelaDTO` e passa para `TViewEditorTabela`.
3. `TViewEditorTabela` coleta os dados e o título, cria/atualiza um `TTabelaDTO` e chama `TCriadorTabelaController.CriarNovaTabela`.
4. `TCriadorTabelaController.CriarNovaTabela`:
   - Pergunta sobre o relatório.
   - Formata nomes e cria diretórios.

- Chama `FXMLService.GravarXML`, `FCSVService.GravarCSV` e `FPDFService.GerarPDF` para persistir os dados.
- Exibe mensagem de sucesso ao usuário.

---

11. Tratamento de Erros

- **Erros Possíveis:**
  - Falha ao criar diretórios (permissões).
  - Falha ao salvar arquivos XML/CSV/PDF (disco cheio, permissões, caminho inválido).
  - Dados inválidos (ex: SQL injection detectado - ver esboço do banco de dados).
  - Formato de arquivo de relatório inválido.
- **Tratamento:**
  - Todos os métodos dos Services (`GravarXML`, `GravarCSV`, `GerarPDF`) devem lançar exceções em caso de erro.
  - O `TCriadorTabelaController.CriarNovaTabela` deve capturar essas exceções e exibir uma mensagem de erro amigável ao usuário usando `ShowMessage`.

---

12. Considerações Futuras/Melhorias

- Suporte completo para cabeçalho na primeira coluna.
- Validação mais robusta de dados (além da detecção de SQL).
- Edição de metadados da tabela.
- Integração com o Banco de Dados conforme esboço (validação, armazenamento de hashes/caminhos).
- Implementação real do `FPDFService` usando uma biblioteca de geração de PDF.

---

13. Informações Adicionais

Exemplo de Arquivo XML Gerado

```
<?xml version="1.0" encoding="UTF-8"?>
<Corpo>
 <Redor>
  <Redor>
   <Tabela>
    <Titulo>Dia1FinanceiroMaio2025</Titulo>
    <Linha numero="1">
     <Colunas>
      <Celula numero="1">Produto</Celula>
      <Celula numero="2">Quantidade</Celula>
      <Celula numero="3">Preço</Celula>
     </Colunas>
```

```
        </Linha>
        <Linha numero="2">
          <Colunas>
            <Celula numero="1">Caneta</Celula>
            <Celula numero="2">100</Celula>
            <Celula numero="3">1.50</Celula>
          </Colunas>
        </Linha>
      </Tabela>
    </Redor>
  </Redor>
</Corpo>
```

Exemplo de Arquivo CSV Gerado

```
"Coluna1","Coluna2","Coluna3"\n
"Produto","Quantidade","Preço"\n
"Caneta","100","1.50"\n
```

Exemplo de Arquivo de Relatório Simples (Reordenação)

```
3,1,2
```

(Este relatório moveria a coluna 3 (Preço) para a primeira posição, a coluna 1 (Produto) para a segunda e a coluna 2 (Quantidade) para a terceira).