Relatório do Código MVC

Estrutura do Código

Modelo (`Cliente`)

Descrição: Representa a entidade `Cliente`, encapsulando seus dados e fornecendo métodos para acessar e modificar esses dados.
Atributos:
nome (String)
endereco (String)
telefone (String)

Métodos Principais:

`getNome()`, `setNome(String nome)`: Obtém e define o nome do cliente.
`getEndereco()`, `setEndereco(String endereco)`: Obtém e define o endereço do cliente.
`getTelefone()`, `setTelefone(String telefone)`: Obtém e define o telefone do cliente.
`toString()`: Retorna uma representação textual do cliente.

Visão (`ClienteView`)

Descrição: Responsável por apresentar as informações do cliente ao usuário.

Métodos Principais:
`mostrarDetalhesCliente(String nome, String endereco, String telefone)`: Exibe os detalhes do cliente, como nome, endereço e telefone.

Controlador (`ClienteController`)

Descrição: Atua como intermediário entre o modelo e a visão. Recebe inputs da visão e atualiza o modelo, além de atualizar a visão com os dados mais recentes do modelo.

Métodos Principais:
`setNomeCliente(String nome)`: Define o nome do cliente no modelo.
`getNomeCliente()`: Obtém o nome do cliente do modelo.

`setEnderecoCliente(String endereco)`: Define o endereço do cliente no modelo.
`getEnderecoCliente()`: Obtém o endereço do cliente do modelo.
`setTelefoneCliente(String telefone)`: Define o telefone do cliente no modelo.
`getTelefoneCliente()`: Obtém o telefone do cliente do modelo.
`atualizarView()`: Atualiza a visão com os dados mais recentes do cliente.

Main(Main)

Descrição: Demonstra o funcionamento do MVC criando instâncias do modelo, visão e controlador, e realizando operações de exemplo.

Principais Ações:
Criação de um cliente com informações iniciais.
Exibição dos detalhes do cliente usando a visão.
Atualização das informações do cliente através do controlador.
Exibição dos detalhes atualizados do cliente.

Benefícios do Padrão MVC

Separação de Responsabilidades: O padrão MVC separa claramente a lógica de negócios (Modelo), a lógica de apresentação (Visão) e a lógica de controle (Controlador). Isso melhora a organização e facilita a manutenção do código.

Facilidade de Manutenção: Alterações na lógica de apresentação não afetam a lógica de negócios, e vice-versa, permitindo alterações e melhorias de forma independente.

Escalabilidade: O padrão permite adicionar novas funcionalidades ou alterar a interface do usuário sem impactar a lógica de negócios ou outros componentes do sistema.

Conclusão

O código apresentado ilustra a aplicação do padrão MVC (Model-View-Controller) para gerenciar clientes em Java. A abordagem MVC separa a lógica de negócios, a interface

do usuário e o controle da aplicação em componentes distintos, promovendo uma organização clara e facilitando a manutenção.

Diagrama