Claro! Os **Cartões CRC** (Classes, Responsabilidades e Colaborações) são uma técnica de design utilizada no desenvolvimento orientado a objetos. Essa abordagem ajuda a identificar e organizar as classes de um sistema, suas responsabilidades, e as colaborações necessárias entre elas para cumprir essas responsabilidades.

Aqui está uma explicação detalhada com exemplos em Java:

**1. O que são Cartões CRC?**

* **Classe**: Representa uma entidade ou conceito no sistema. Por exemplo, Usuário, Pedido, ou Produto.
* **Responsabilidades**: Definem o que a classe deve fazer. Geralmente, são métodos ou comportamentos relacionados ao propósito da classe.
* **Colaborações**: Indicam com quais outras classes a classe atual precisa interagir para cumprir suas responsabilidades.

Cada cartão CRC tem três seções principais:

1. Nome da **Classe**.
2. Lista de **Responsabilidades**.
3. Lista de **Colaborações**.

**2. Exemplo com Cartões CRC**

Vamos modelar um sistema simples de **Pedido de Compras Online**. Suponha que temos três classes principais: Cliente, Pedido, e Produto.

**Classe: Cliente**

* **Responsabilidades**:
  + Armazenar informações do cliente (nome, email).
  + Fazer pedidos.
* **Colaborações**:
  + Colabora com a classe Pedido para criar um novo pedido.

package crc\_testejunit;

public class Cliente {

private String nome;

private String email;

public Cliente(String nome, String email) {

this.nome = nome;

this.email = email;

}

public Pedido fazerPedido(Produto produto, int quantidade) {

return new Pedido(this, produto, quantidade);

}

// Getters e Setters

public String getNome() {

return nome;

}

public void setNome(String nome) {

this.nome = nome;

}

public String getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

}

**Classe: Pedido**

* **Responsabilidades**:
  + Armazenar informações do pedido (cliente, produto, quantidade).
  + Calcular o valor total do pedido.
* **Colaborações**:
  + Colabora com a classe Produto para obter o preço do produto.
  + Colabora com a classe Cliente para identificar o cliente que fez o pedido.

package crc\_testejunit;

public class Pedido {

private final Cliente cliente; // final para indicar que não será modificado

private final Produto produto; // final porque o produto não muda após ser definido

private final int quantidade;

public Pedido(Cliente cliente, Produto produto, int quantidade) {

this.cliente = cliente;

this.produto = produto;

this.quantidade = quantidade;

}

public double calcularTotal() {

return produto.getPreco() \* quantidade;

}

// Apenas getters para acessar os dados, sem setters

public Cliente getCliente() {

return cliente;

}

public Produto getProduto() {

return produto;

}

public int getQuantidade() {

return quantidade;

}

}

**Classe: Produto**

* **Responsabilidades**:
  + Armazenar informações sobre o produto (nome, preço).
* **Colaborações**:
  + Nenhuma colaboração direta neste caso, pois é uma classe independente que fornece dados para outras classes.

package crc\_testejunit;

public class Produto {

private String nome;

private double preco;

public Produto(String nome, double preco) {

this.nome = nome;

this.preco = preco;

}

public double getPreco() {

return preco;

}

public void setPreco(double preco) {

this.preco = preco;

}

public String getNome() {

return nome;

}

public void setNome(String nome) {

this.nome = nome;

}

}

**3. Representação dos Cartões CRC**

**Cartão CRC do Cliente**

* **Classe**: Cliente
* **Responsabilidades**:
  1. Armazenar informações do cliente.
  2. Fazer pedidos.
* **Colaborações**:
  1. Pedido.

**Cartão CRC do Pedido**

* **Classe**: Pedido
* **Responsabilidades**:
  1. Armazenar informações do pedido.
  2. Calcular o valor total.
* **Colaborações**:
  1. Cliente.
  2. Produto.

**Cartão CRC do Produto**

* **Classe**: Produto
* **Responsabilidades**:
  1. Armazenar informações do produto (nome e preço).
* **Colaborações**:
  1. Nenhuma.

**4. Como usar Cartões CRC no Design de Sistemas?**

1. Identifique as classes principais no sistema.
2. Defina claramente as responsabilidades de cada classe.
3. Determine quais classes precisam colaborar para cumprir essas responsabilidades.
4. Use os cartões CRC como base para implementar as classes no código.

**5. Benefícios dos Cartões CRC**

* Promovem um design modular e claro.
* Facilita a identificação de responsabilidades e evita sobrecarga em uma única classe.
* Melhoram a comunicação na equipe de desenvolvimento.

Essa abordagem pode ser muito útil para planejar sistemas de médio ou grande porte antes de começar a codificação.

Uma imagem com texto, captura de ecrã, software, Software de multimédia

Descrição gerada automaticamente

Uma imagem com texto, captura de ecrã, software, Tipo de letra

Descrição gerada automaticamente

Uma imagem com texto, captura de ecrã, software, Tipo de letra

Descrição gerada automaticamente

Uma imagem com texto, captura de ecrã, Tipo de letra

Descrição gerada automaticamente