Aula 13 - Classe Abstrata

Uma classe abstrata é desenvolvida para representar entidades e conceitos abstratos, sendo utilizada como uma classe pai, pois não pode ser instanciada. Ela define um modelo (template) para uma funcionalidade e fornece uma implementação incompleta - a parte genérica dessa funcionalidade - que é compartilhada por um grupo de classes derivadas. Cada uma das classes derivadas **completa a funcionalidade da classe abstrata** adicionando um comportamento específico.

Uma classe abstrata normalmente possui métodos abstratos. Esses métodos são implementados nas suas classes derivadas concretas com o objetivo de definir o comportamento específico. **O método abstrato define apenas a assinatura do método** e, portanto, não contém código.

Uma classe abstrata pode também possuir atributos e métodos implementados, componentes estes que estarão integralmente acessíveis nas subclasses, a menos que o mesmo seja do tipo private.

Entretanto, a classe abstrata possui algumas características diferentes, são elas:

1. Não suporta herança múltipla;

2. Pode conter métodos implementados ou abstratos: Isso quer dizer que todos os métodos de uma classe abstrata podem ser tanto concretos como, também, todos podem ser abstratos;

3. Pode conter atributos de todos os tipos;

4. Pode conter construtor;

5. Não pode ser instanciada;

Uma classe abstrata deve ser utilizada quando queremos criar várias classes que irão compartilhar um mesmo comportamento, uma classe abstrata é o componente ideal para ser a base para criação de todas elas, servindo como um molde para as futuras classes que irão derivar dela.

Sendo assim, classes abstratas definem a identidade de suas classes derivadas ditando o que e como uma classe deve se comportar, o que aumenta o acoplamento entre classes, porém faz total sentido em algumas situações.

Vamos ver um exemplo:

Conta.java

```
public abstract class Conta {
    private String agencia;
    private String numero;
```

```java
    private double saldo;

    public String getAgencia() {
        return agencia;
    }

    public void setAgencia(String agencia) {
        this.agencia = agencia;
    }

    public String getNumero() {
        return numero;
    }

    public void setNumero(String numero) {
        this.numero = numero;
    }

    public double getSaldo() {
        return saldo;
    }

    public void setSaldo(double saldo) {
        this.saldo = saldo;
    }

    protected String exibirMensagem() {
        return "Bem vindo ao Internet Banking do Prof. Joseffe!";
    }

    Conta(){

    }

    Conta(String agencia, String numero, double salario){
        this.agencia = agencia;
        this.numero = numero;
        this.saldo = salario;
    }

    public abstract void Sacar()
```

```
    public void Depositar(double valor) {
        this.saldo += valor;
    }

    public void Depositar(double valor, String nomeDepositante) {
        this.saldo += valor;
    }

    public void Depositar(String nomeDepositante, double valor, String
telefoneDepositante) {
        this.saldo += valor;
    }
}
```

Desenvolvemos a classe Conta (super classe) como uma classe abstrata. Ou seja, ela não pode ser instanciada. Além disso, incluímos um método “Sacar” abstrato, sem implementação.

ContaCorrente.java

```
public class ContaCorrente extends Conta{
    private double chequeEspecial;

    public double getChequeEspecial() {
        return chequeEspecial;
    }

    public void setChequeEspecial(double chequeEspecial) {
        this.chequeEspecial = chequeEspecial;
    }

    public ContaCorrente() {
    }

    public ContaCorrente(String agencia, String numero, double salario,
double chequeEspecial) {
        super(agencia, numero, salario);

        this.chequeEspecial = chequeEspecial;
    }

    public void Depositar(double valor) {
        super.Depositar(valor);
```

```
        valor = valor - 0.10;
        this.setSaldo(valor);
    }
}
```

ContaPoupanca.java

```
public class ContaPoupanca extends Conta{
    private double rentabilidade;

    public double getRentabilidade() {
        return rentabilidade;
    }

    public void setRentabilidade(double rentabilidade) {
        this.rentabilidade = rentabilidade;
    }

    public ContaPoupanca() {
    }

    public ContaPoupanca(String agencia, String numero, double salario,
double rentabilidade) {
        super(agencia, numero, salario);

        this.rentabilidade = rentabilidade;
    }

    public void Depositar(double valor) {
        super.Depositar(valor);

        valor = valor + 0.50;
        this.setSaldo(valor);
    }
}
```

Como as classes ContaCorrente e ContaPoupanca herdam da classe Conta e a classe Conta é uma classe abstrata e possui um método abstrato, é obrigatório a implementação do método Sacar nas sub classes ContaCorrente e ContaPoupanca.