

Mãos na massa: Focando em herança e polimorfismo

Chegou o momento de implementarmos o que foi visto em aula

- 1) Crie um novo projeto, continuaremos com o nome **bytebank-herdado-conta**.
- 2) Copie as duas classes (Cliente e Conta) do projeto anterior e coloque na pasta **src** do novo projeto
- 3) Clique com o botão direito no projet atual e selecione a opção **"Close unrelated projects"** para fecharmos tudo que não for relacionado ao projeto atual.
- 4) Crie uma nova classe, no curso utilizamos o nome de **ContaCorrente**, repare que na hora da criação, logo abaixo podemos definir a **super classe**, clique em **Browse**, escreva e selecione a classe **Conta**
- 5) Escreva o construtor da classe **ContaCorrente**, repassando os parâmetros para a classe mãe através so **super()**;

```
public ContaCorrente(int agencia, int numero){  
    super(agencia, numero);  
}
```

- 6) Crie agora a classe **ContaPoupanca** já definindo a classe **Conta** como mãe.
- 7) Crie um construtor da mesma forma que foi feito na classe **ContaCorrente**:

```
public ContaPoupanca(int agencia, int numero){  
    super(agencia, numero);  
}
```

- 8) Crie uma classe de teste, no curso, demos o nome de **TesteContas**, essa classe deve ter o método **main**!
- 7) Dentro do método **main**, instancie uma conta corrente e uma poupança e então realize algumas transações, veja o código abaixo:

```
//Dentro do método main  
  
ContaCorrente cc = new ContaCorrente(111, 111);  
cc.deposita(100.0);  
  
ContaPoupanca cp = new ContaPoupanca(222, 222);  
cp.deposita(100.0);
```

- 8) Utilize o método transfere para transferir valores de uma conta para a outra:

```
//Dentro do método main  
  
ContaCorrente cc = new ContaCorrente(111, 111);  
cc.deposita(100.0);
```

```
ContaPoupança cp = new ContaPoupança(222, 222);
cp.deposita(100.0);

cc.transfere(10.0, cp);
System.out.println("CC: " + cc.getSaldo());
System.out.println("CP: " + cp.getSaldo());
```

9) Comente os **System.out.println** do construtor execute o nosso **TesteContas** e veja se as transações foram feitas com sucesso.

OBS: Na classe **Conta**, iniciamos as contas com o saldo de 100 (no construtor), comente essa linha para não confundir!

10) Na classe **ContaCorrente**, sobrescreva o método **saca**, para isso, escreva **saca** e utilize o atalho do eclipse **CTRL + ESPAÇO** e selecione a opção que contenha **Override method**.

11) Implemente a nova regra de negócio, o código deve ficar conforme abaixo:

```
@Override
public boolean saca(double valor){
    double valorASacar = valor + 0.2;
    return super.saca(valorASacar);
}
```

12) Execute novamente e veja se tudo saiu conforme o planejado!