12

Mão na massa: Generics

Chegou a hora de você pôr em prática o que foi visto na aula. Para isso, execute os passos listados abaixo.

1) Para demonstrar problemas no `ArrayList`, adicione um objeto do tipo `Cliente` e execute a classe `Teste`. Você verá que vai ocorrer uma exceção do tipo `ClassCastException` porque não foi possível converter a referência do tipo `Cliente` em uma `Conta`.

```
package br.com.bytebank.banco.test.util;

import java.util.ArrayList;

public class Teste {
    public static void main(String[] args) {

        ArrayList lista = new ArrayList();

        //Conta cc = new ContaCorrente(22, 11);
        Cliente cliente = new Cliente();
        lista.add(cliente);

        Conta cc2 = new ContaCorrente(22, 22);
        lista.add(cc2);

        System.out.println("Tamanho: " + lista.size());

        Conta ref = (Conta) lista.get(0);
        System.out.println(ref.getNumero());

        lista.remove(0);
        System.out.println("Tamanho: " + lista.size());

        Conta cc3 = new ContaCorrente(33, 311);
        lista.add(cc3);

        Conta cc4 = new ContaCorrente(33, 322);
        lista.add(cc4);

        for(int i = 0; i < lista.size(); i++) {
            Object oRef = lista.get(i);
            System.out.println(oRef);
        }

        System.out.println("----------");

        for(Object oRef : lista) {
            System.out.println(oRef);
        }

    }
}
```

2) Informe ao compilador que você só deseja criar um array de contas. Modifique a classe `Teste` para isso:

```
package br.com.bytebank.banco.test.util;

import java.util.ArrayList;

public class Teste {
    public static void main(String[] args) {

        ArrayList<Conta> lista = new ArrayList<Conta>();

        Conta cc = new ContaCorrente(22, 11);
        lista.add(cc);

        Conta cc2 = new ContaCorrente(22, 22);
        lista.add(cc2);

        System.out.println("Tamanho: " + lista.size());

        Conta ref = (Conta) lista.get(0);
        System.out.println(ref.getNumero());

        lista.remove(0);
        System.out.println("Tamanho: " + lista.size());

        Conta cc3 = new ContaCorrente(33, 311);
        lista.add(cc3);

        Conta cc4 = new ContaCorrente(33, 322);
        lista.add(cc4);

        for(int i = 0; i < lista.size(); i++) {
            Object oRef = lista.get(i);
            System.out.println(oRef);
        }

        System.out.println("----------");

        for(Object oRef : lista) {
            System.out.println(oRef);
        }

    }
}
```

3) Experimente incluir um objeto de outro tipo na lista acima. O compilador não vai deixar!

4) Modifique seu código para aproveitar os benefícios do generics, removendo o *cast* no método `get()` e ao usar o *enhanced for*:

```
package br.com.bytebank.banco.test.util;

import java.util.ArrayList;

public class Teste {
```

```
    public static void main(String[] args) {

        ArrayList<Conta> lista = new ArrayList<Conta>();

        Conta cc = new ContaCorrente(22, 11);
        lista.add(cc);

        Conta cc2 = new ContaCorrente(22, 22);
        lista.add(cc2);

        System.out.println("Tamanho: " + lista.size());

        Conta ref = lista.get(0);
        System.out.println(ref.getNumero());

        lista.remove(0);
        System.out.println("Tamanho: " + lista.size());

        Conta cc3 = new ContaCorrente(33, 311);
        lista.add(cc3);

        Conta cc4 = new ContaCorrente(33, 322);
        lista.add(cc4);

        for(int i = 0; i < lista.size(); i++) {
            Object oRef = lista.get(i);
            System.out.println(oRef);
        }

        System.out.println("----------");

        for(Conta oRef : lista) {
            System.out.println(oRef);
        }

    }
}
```

5) Ótimo! Agora seu código ficou mais seguro e expressivo com o uso do **Generics**!