



Usando interfaces sempre que possível

Perceba que, até agora, usamos sempre a classe concreta diretamente como tipo das variáveis que guardavam nossas coleções (`ArrayList` , `HashMap` etc.). Por exemplo, na classe `Pagamentos` , temos:

```
private ArrayList<Pagamento> pagamentos = new ArrayList<Pagamento>();
```

E, como retorno dos métodos, temos:

```
public ArrayList<Pagamento> pagamentosAntesDe(Calendar data) {...}
```

Será que é uma boa fixarmos `ArrayList` ? E se resolvermos trocar o `ArrayList` por `HashSet` , o que acontece com quem usa nossa classe `Pagamentos` ? Teremos que corrigir todas elas! Veja que estamos acoplando quem usa nossa classe a um tipo específico de coleção que usamos para guardar nossos pagamentos.

Vamos diminuir esse acoplamento. A classe `ArrayList` implementa uma interface que representa coleções: a `Collection` . Troque o tipo da variável `pagamentos` e os retornos dos métodos que devolvem `ArrayList` para usar essa interface.

Cole aqui o código refatorado.

Responda

INSERIR CÓDIGO	FORMATAÇÃO
<div><pre>...</pre></div>	