Certified Tech Developer
The Ultimate Degree

Generics

Programação paramétrica

A programação paramétrica não é exclusiva do Java. Na verdade, a linguagem demorou muito para incorporá-lo. Ada, uma linguagem orientada a objetos, tem contemplado isso desde os anos oitenta. Existem outras linguagens que incorporam esse recurso, como C ++ (por meio de modelos), D, Eiffel, Delphi, TypeScript, etc. Existem muitas outras linguagens que usam "tipos de dados paramétricos". Alguns chamam de genericidade, uma vez que um tipo de dado genérico é estabelecido para a estrutura ou classe de dados que está sendo definida, que será estabelecida no momento do uso. Em Java, é chamado da mesma forma: **Generics.**

Simplificando, **Generics** é sobre adiar a pergunta "Que tipo é este objeto?" Assim, o "tipo" do objeto fica como parâmetro que o programador irá definir ao trabalhar com aquele objeto.

Sintaxe e uso

Em Java, para definir um tipo de dado paramétrico, usamos o "<>". Vamos dar um exemplo simples: um balde. Pode conter muitas coisas, sujeira, areia, água, combustível, etc.

Vamos ver como colocar isso no código:

```
public class Balde<T> {
   private T conteudo;

   public Balde() {
   }

   public preencher(T conteudo) {
       this.conteudo = conteudo;
   }

   public T obterConteudo() {
       return conteudo;
   }
}
```

Aqui, quando definimos a classe balde, não definimos o tipo de seu conteúdo (o tipo de seu atributo “conteúdo”), mas “adiamos” essa decisão até o momento de usá-lo. Ou seja, deixamos o tipo como parâmetro. Este parâmetro é definido pela letra T (pode ser qualquer letra do alfabeto). Normalmente T é usado para tipo. Isso também acontece com operações sobre conteúdo: limitamos seu uso de acordo com o tipo de atributo de conteúdo.

Agora, supondo que temos as classes água, areia, combustível, etc., poderíamos fazer:

```
Agua a = new Agua();
Combustivel c = new Combustivel();
Balde<Agua> b = new Balde<>();
b.setConteudo(a);

// NÃO COMPILA! Não posso colocar combustível em um balde d'água!
//b.setConteudo(c);

// se o balde é de água, eu sempre terei água nele
System.out.println(“Vou tomar” + b.obterConteudo());
```

Como podemos ver, não existiam casts para operar o conteúdo, portanto, não havia perigo de erros no momento da execução. Além disso, não podemos colocar algo que não corresponda no balde.

Então, por que não tornamos o tipo de conteúdo "Objeto"? Isso nos permitiria colocar água, combustível ou areia no balde...

```
public class Balde {
   private Object conteudo;

   public Balde() {
   }

   public preencher(Object conteudo) {
       this.conteudo = conteudo;
   }

   public Object obtenerConteudo() {
       return conteudo;
   }
}
```

É totalmente válido, mas não devemos apenas lançar, mas poderíamos misturar o conteúdo do balde:

```
   Agua a = new Agua();
   Combustivel c = new Combustivel();
   Balde b = new Balde ();
   b.setConteudo(a);
   b.setConteudo(c);

  //Duvido que o conteúdo tenha um sabor agradável ...
  System.out.println("Vou tomar" + b.obterConteudo());
```

É por isso que o uso de tipos paramétricos, ou seja, Genéricos, torna-se interessante.