DIN Departamento de Informática

UEM Universidade Estadual de Maringá

Interfaces em Java

Lilian Passos Scatalon
lpscatalon2@uem.br

Interface

Mecanismo para especificar o comportamento de uma classe

Se uma classe implementa uma interface, ela assume o compromisso de apresentar o comportamento declarado na interface

Serve para estabelecer um contrato entre uma classe e o código que vai usá-la

Declarando uma interface

Usa-se a palavra-chave interface e é preciso providenciar os cabeçalhos dos métodos

```
public interface IntSequence {
        boolean hasNext();
        int next();
}
```

Todos os métodos da interface são automaticamente públicos

É possível declará-los explicitamente com public para maior clareza

Usando uma interface

Para o código que usa uma interface é suficiente saber apenas o que é declarado nela

```
public static double average(IntSequence seq, int n) {
        int count = 0;
        double sum = 0;
        while (seq.hasNext() && count < n) {
                count++;
                sum += seq.next();
        }
        return count == 0 ? 0 : sum / count;
}
```

Implementando uma interface

Considerando o exemplo, uma sequência de inteiros pode tomar diversas formas:

- Uma sequência de inteiros aleatórios
- A sequência de números primos
- A sequência de elementos de um vetor

A interface IntSequence fornece um único mecanismo para lidar com todos esses tipos de sequência

Implementando uma interface

Uma classe que implementa IntSequence deve implementar os métodos declarados na interface

```
public class SquareSequence implements IntSequence {

        private int i;
        public boolean hasNext() {
                return true;
        }

        public int next() {
                i++;
                return i * i;
        }
}
```

Usando uma interface

```
public static double average(IntSequence seq, int n) {
        int count = 0;
        double sum = 0;
        while (seq.hasNext() && count < n) {
                count++;
                sum += seq.next();
        }
        return count == 0 ? 0 : sum / count;
}
```

```
SquareSequence squares = new SquareSequence();
double avg = average(squares, 100);
```

Implementando uma interface

A classe DigitSequence também implementa IntSequence

```
public class DigitSequence implements IntSequence {

        private int number;

        public DigitSequence(int n) {
                number = n;
        }

        public boolean hasNext() {
                return number != 0;
        }
        public int next() {
                int result = number % 10;
                number /= 10;
                return result;
        }

        public int rest() {
                return number;
        }

}
```

Interfaces em Java

Mecanismo de abstração

Java não permite herança múltipla, mas o conceito de interfaces em Java contorna essa restrição

Uma classe pode herdar de apenas uma subclasse, mas pode implementar muitas interfaces

Interface em Java também representa o relacionamento é-um

- Exemplo: a sequência de quadrados (SquareSequence) *é uma* sequência de inteiros (IntSequence)

Interfaces em Java

Interfaces especificam o que a classe deve fazer e não como

Não é possível instanciar um objeto de uma interface

Geralmente uma interface não contém métodos concretos, ou seja com implementação (a partir do Java 8 é possível ter uma implementação *default*)

Na interface usa-se o modificador de acesso public

Convertendo para o tipo da interface

```
IntSequence digits = new DigitSequence(1729);
double avg = average(digits, 100);
// vai computar apenas os quatro primeiros valores da sequência
```

A variável digits tem o tipo IntSequence (a interface)

Uma variável do tipo IntSequence referencia um objeto de alguma classe que implementa a interface IntSequence

Tipo e Supertipo

Um tipo S é um supertipo de um tipo T (o subtipo) quando qualquer valor do subtipo pode ser atribuído a uma variável do supertipo sem conversão

Por exemplo, a interface IntSequence é um supertipo da classe DigitSequence

Casts

Quando é necessário fazer o contrário, converter de um supertipo para um subtipo, usa-se um cast

```
IntSequence sequence = ...;
DigitSequence digits = (DigitSequence) sequence;
System.out.println(digits.rest());
```

Aqui o cast é necessário porque rest é um método de DigitSequence, mas não da interface IntSequence

Casts

Só é possível fazer o cast de um objeto para a sua própria classe ou para algum de seus supertipos

```
IntSequence sequence = ...;
String digitString = (String) sequence;
// Erro – IntSequence não é um supertipo de String
```

Operador instanceof

Para evitar problemas, é possível testar se um objeto é de um determinado tipo, usando o operador instanceof

```
object instanceof Type
```

Essa expressão retorna true se object é instância de uma classe que tenha Type como supertipo

Operador instanceof

É uma boa ideia fazer essa verificação antes de usar um cast

```
if (sequence instanceof DigitSequence) {
    DigitSequence digits = (DigitSequence) sequence;
    ...
}
```

Estendendo interfaces

Uma interface pode estender outra, usando o mesmo mecanismo de herança

```
public interface Closeable {
    void close();
}

public interface Channel extends Closeable {
    boolean isOpen();
}
```

Uma classe que implemente Channel deve providenciar ambos os métodos e os respectivos objetos podem ser convertidos aos dois tipos de interface

Implementando múltiplas interfaces

Uma classe pode implementar qualquer número de interfaces

```
public class FileSequence implements IntSequence, Closeable {
  ...
}
```

Constantes

Qualquer variável definida em uma interface é automaticamente public static final

```
public interface SwingConstants {
        int NORTH = 1;
        int NORTH_EAST = 2;
        int EAST = 3;
        ...
}
```

Bibliografia

H. Schildt. **Java: A Beginner's Guide**. McGraw-Hill Education, 8th edition, 2018.

C. S. Horstmann. **Core Java SE 9 for the Impatient**. Addison-Wesley, 2nd Edition, 2018.

Caelum. Apostila **Java e Orientação a Objetos**. Disponível em https://www.alura.com.br/apostila-java-orientacao-objetos