Domain Driven Design

PROF. ELIANE RODRIGUES MARION SANTA ROSA
Profeliane.rosa@fiap.com.br

1

INTERFACE

INTERFACE

Uma interface pode definir uma série de métodos, mas nunca conter implementação deles. Ela só expõe **o que o objeto deve fazer**, e não como ele faz, nem o que ele tem. **Como ele faz** vai ser definido em uma implementação dessa interface.

Sintaxe:

- Uma interface é declarada com o comando interface;
- Todos os atributos e métodos de uma interface devem ser públicos;
- Todo atributo em uma interface é implicitamente final e estático;
- Todo método em uma interface é abstrato, por isso não é preciso declarar o public e nem o abstract.

INTERFACE

A vantagem principal das interfaces é que não há limites de quantas interfaces uma classe pode implementar. O que ajuda no caso de heranças múltiplas que não é possível ser feito em Java, pois uma classe apenas pode herdar as características de uma outra classe.

Uma interface é criada da mesma forma que uma classe, mas utilizando a palavra-chave interface no lugar de class.

INTERFACE

```
interface NomeInterface {

    //campo static e final
    static final int nomeCampo = 10;

    //método abstrato
     void nomeMetodo();

}
```

INTERFACE

- Uma interface **NÃO** é construída para ser executada e nem instanciada.
- Só há um modo de instanciar um objeto cujo tipo seja uma interface: utilizar um construtor de uma classe que a tenha incluído em sua declaração implements e tenha implementado todos os seus métodos.
- A responsabilidade por implementar os métodos, cujas assinaturas estão na interface será das classes;
- O comando **implements** é utilizado para indicar que uma classe irá implementar os métodos de uma interface.

PALAVRA CHAVE IMPLEMENTS

FIAP

```
public class NomeClasse implements NomeInterface {

    @Override
    public void nomeMetodo() {
        // deve implementar
    }

}
```

INTERFACE

FIAP

- Diferentemente das classes, uma interface pode herdar de mais de uma interface. Em interface, as classes podem expandir seus próprios tipos pela implementação de uma ou mais interfaces.

- É como um **contrato** que depende que outros contratos sejam fechados antes deste valer. Você não herda métodos e atributos, mas sim responsabilidades.

- Ela é um contrato onde quem assina se responsabiliza por implementar esses métodos (cumprir o contrato).

2

POLIMORFISMO

POLIMORFISMO

- Termo definido em linguagens orientadas a objeto, como por exemplo Java, C# e C++, que permite ao desenvolvedor usar o mesmo elemento de formas diferentes.

- É a capacidade de um objeto poder ser referenciado de várias formas, sendo que as funcionalidades são utilizadas de forma dinâmica por um programa no decorrer de sua execução.

- Apresenta uma situação na qual um objeto pode se comportar de maneiras diferentes ao receber uma mensagem.

- Os mesmos atributos e objetos podem ser utilizados em objetos distintos, porém, com implementações lógicas diferentes.

POLIMORFISMO

- http://www.dsc.ufcg.edu.br/~jacques/cursos/p2/html/oo/o_que_e_polimorfismo.htm