PROGRAMAÇÃO ORIENTADA A OBJETOS

ENCAPSULAMENTO

AGENDA

1. Conceito de Encapsulamento.
2. Interface *versus* implementação.
3. Características do Encapsulamento.

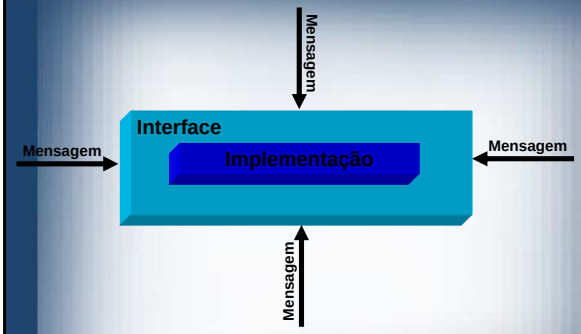
ENCAPSULAMENTO

- **Encapsular** significa combinar dados e comportamento e um pacote e ocultar a implementação dos dados do usuário do objeto.

ENCAPSULAMENTO

- **Encapsulamento** é a característica da OO de ocultar partes independentes da implementação.
  - Permite que o sistema possa ser dividido em várias partes;
  - Cada parte realiza o seu trabalho independente das outras;
  - Cada parte oculta seus detalhes/funcionamento interno.

ENCAPSULAMENTO



INTEFACE x IMPLEMENTAÇÃO

- Interface:
  - Lista os serviços fornecidos por um componente;
  - Contrato do componente com o mundo exterior, que define exatamente o que uma entidade externa pode fazer para o objeto;
  - Painel de controle do objeto.

INTERFACE X IMPLEMENTAÇÃO

- Implementação
  - Define como um componente fornece um serviço;
  - Define os detalhes internos do componente.

INTEFACE x IMPLEMENTAÇÃO

Implementação

```
public class Log {
   public void debug (String message) {
      System.out.println ("DEBUG: " +
                                 message);
   }
   public void info (String message) {
      System.out.println ("INFO: " +
                                 message);
   }
   public void warning (String message)
   {
      System.out.println ("WARNING: " +
                                 message);
   }
   public void error (String message) {
      System.out.println ("ERROR: " +
                                 message);
   }
   public void fatal (String message) {
      System.out.println ("FATAL: " +
                                 message);
   }
}
```

Interface

```
public void debug (String message)
public void info (String message)
public void warning (String message)
public void error (String message)
public void fatal (String message)
```

Mensagens

```
Log log = new Log();
log.debug("mensagem");
log.error("messagem");
log.info("messagem");
```

Resultado

```
DEBUG: mensagem
ERROR: messagem
INFO: messagem
```

INTEFACE x IMPLEMENTAÇÃO

Implementação

```
public class Log {
   public void debug (String message) {
      print ("DEBUG", message);
   }
   public void info (String message) {
      print ("INFO", message);
   }
   public void warning (String message) {
      print ("WARNING", message);
   }
   public void error (String message) {
      print ("ERROR", message);
   }
   public void fatal (String message) {
      print ("FATAL", message);
   }
   private void print (String severity, String message) {
      System.out.println (severity + ": " + message);
   }
}
```

Interface

```
public void debug (String message)
public void info (String message)
public void warning (String message)
public void error (String message)
public void fatal (String message)
```

Mensagens

```
Log log = new Log();
log.debug("mensagem");
log.error("messagem");
log.info("messagem");
```

Resultado

```
DEBUG: mensagem
ERROR: messagem
INFO: messagem
```

INTERFACE x IMPLEMENTAÇÃO

- Outros exemplos:
  - Métodos públicos de Carro/Motorista e implementação de ambos.
  - Métodos públicos de Televisao/Controle e implementação de ambos.

INTERFACE x IMPLEMENTAÇÃO

- Cabe aqui ressaltar que a palavra **interface** em Java tem três significados:
  - **Interface** Gráfica do Usuário (GUI);
  - **Interface** da classe;
  - Uma estrutura semelhante a uma classe chamada **interface**.

CARACTERÍSTICAS DO ENCAPSULAMENTO

- O encapsulamento transforma os objetos em componentes plugáveis;
- Isso traz três vantagens:
  - **Independência**: os objetos podem ser utilizados em qualquer lugar;
  - **Transparência**: enquanto a interface não for alterada todas as alterações internas não irão acarretar a troca de outros objetos;
  - **Sem efeitos colaterais**: um objeto só terá as interações definidas pela interface.

CARACTERÍSTICAS DO ENCAPSULAMENTO

- As principais características do encapsulamento são:
  - **Abstração:**
    - Pensar e programar de forma abstrata;
  - **Ocultação da Implementação:**
    - Guardar os segredos;
  - **Divisão de responsabilidade:**
    - Preocupar-se com seus próprios negócios.

CARACTERÍSTICA 1: ABSTRAÇÃO

- **Pensar e programar de forma abstrata.**
- O que é abstração?
  - Abstração pode ser visto também como o processo de simplificar um problema difícil;
- Vantagens
  - Permite que resolva um problema facilmente;
  - Permite que uma solução semelhante possa ser reutilizada.

CARACTERÍSTICA 2: OCULTAÇÃO DA IMPLEMENTAÇÃO

- **Guardar os segredos.**
- A ocultação da implementação tem duas vantagens:
  - Protege os objetos dos usuários;
  - Protege os usuários dos objetos do próprio objeto.

CARACTERÍSTICA 03: DIVISÃO DE RESPONSABILIDADE

- **Preocupar-se com seus próprios negócios.**
  - Para se ter um código **fracamente acoplado**, deve-se ter uma divisão de responsabilidade correta;
  - Cada objeto deve executar uma função;
  - Todas as funcionalidades devem trabalhar no sentido de uma responsabilidade comum.