JOVEM PROGRAMADOR

UMA PROPOSTA INOVADORA

SENAC - ARARANGUÁ

Desenvolvimento de sistemas com POO

Você já aprendeu como criar um objeto a partir de uma classe. Vamos lembrar como ficou o código nos próximos slides :

```
//arquivo UmObjeto.java
public class UmObjeto {
  private String nome;
  private double tamanho;

  public UmObjeto() {
  }
  public UmObjeto(String nome, double tamanho){
    this.nome = nome;
    this.tamanho = tamanho;
  }
  public void setNome (String nome) {
    this.nome = nome;
  }
```

(continua...)

(...continuação)

```
  public void setTamanho(double tamanho) {
    this.tamanho = tamanho;
  }
  public String getNome() {
    return nome;
  }
  public double getTamanho() {
    return tamanho;
  }
}
```

```
//arquivo Test.java
public class Test {
  public static void main(String []args) {
    UmObjeto lapis = new UmObjeto();

    // parâmetros via método set
    lapis.setNome("lápis");
    lapis.setTamanho(15.5);

    // pegando os valores com get
    System.out.println (lapis.getNome());
    System.out.println (lapis.getTamanho());
  }
}
```

O que significa algo público?

Se você respondeu : “Oras, é para que TODOS tenham acesso!”, está com toda a razão. Vamos pensar em um caso especial para ficar bem claro.

Um restaurante não nega água às pessoas!

```
public class Restaurante {
  public String agua; // A água será acessível a todos
}
```

Mas como fica o acesso ao buffet? Precisa verificar o pagamento?

Se fizermos isso, as crianças de colo também teriam que pagar. Os funcionários também teriam que pagar. Ninguém teria acesso direto fora da classe. Só se passasse por um método de verificação de alguma condição!

O acesso direto ao buffet está bloqueado!

```
public class Restaurante {
  public String agua; // A água será acessível a todos
  private String buffet; // Sem acesso direto!
}
```

Olha o "encapsulamento"!

Apenas quem passar por uma condição terá acesso ao buffet. "Encapsulamos" o atributo com *private*. E isso será resolvido a partir de método público de acesso que terá algum *if,* concorda?! Temos que pensar nisso.

E se quisermos que o acesso seja só dentro do pacote?

Para as crianças de colo podemos liberar o acesso ao buffet, mas elas só podem ser servidas pelos seus pais. Ou seja, se é filho, o acesso não é público mas protegido ou *protected*. Agora a regra muda e bastará estar dentro do pacote (o restaurante!) para ter acesso ao buffet.

Com *protected* todos no pacote podem acessar, inclusive os herdeiros

Atributos *protected* ficam restritos ao pacote. Além disso, se alguém estender a classe, continuaria a ter acesso ao atributo.

A diferença entre *public* e *protected*, neste caso, é que a água estará liberada mesmo que a classe Restaurante seja importada para outro pacote. O buffet não!

Agora o buffet tem uma restrição de acesso menor

```
public class Restaurante {
  public String agua; // A água será acessível a todos
  protected String buffet; // Acessível no restaurante
}
```

E as bebidas alcoólicas? Só para adultos!

Agora sim! Desta vez a restrição é muito importante! Pode até estar dentro do restaurante, mas só poderá beber álcool se for adulto. A restrição do acesso está justificada!

A bebida será, de fato, privativa na condição da idade!

```
public class Restaurante {
  public String agua; // A água será acessível a todos
  protected String buffet; // Só estar no restaurante!
  private String bebida; // Precisa ser um adulto
}
```

Como ficou o nosso código?

Vamos ver as restrições na prática! Pelo que vimos, nosso código deverá ter dois pacotes. Um é o "restaurante" e o outro poderia ser "teste". Haverá um *import* da classe e a instanciação de um objeto do tipo Restaurante para verificarmos os bloqueios.

```
//arquivo restaurante.java
package restaurante;
public class Restaurante {
  public String agua;
  protected String buffet;
  private String bebida;

  public void servirBebida (boolean beberAlcool) {
    if(beberAlcool) {
       this.bebida = “Bebida servida”;
    } else {
         this.bebida = “Acesso somente a adultos!”;
    }
    System.out.println(this.bebida);
  }
```

(continua...)

(...continuação)

```
  public void acessarBuffet {
    this.buffet = "Acesso liberado";
    System.out.println(this.buffet);
  }
}

//arquivo Test.java
package teste;
import restaurante.Restaurante;
public class Test {
  public static void main(String []args) {
    Restaurante r = new Restaurante();
```

(continua...)

(...continuação)

```
// Acesso direto
    r.agua = “Agua servida”;
    System.out.println(r.agua);

// Bloqueios
// r.buffet e r.bebida não estão acessíveis!
// Somente via métodos públicos ou seja:

    r.acessarBuffet();
    r.servirBebida(true);

  }
}
```