

Modificadores de acesso em Java

Disciplina: Desenvolvimento de Aplicações



Conteúdos:

Modificadores de acesso em Java.

Bloco 1

Qual é o seu humor hoje?

Qual é a escultura que mais te representa?



1



2



3



4



5



6

Modificadores de acesso

Modificadores de acesso são palavras-chave em Java que determinam a visibilidade e o acesso de classes, métodos e atributos em diferentes partes do código. Eles ajudam no controle de visibilidade, pois controlam quem pode acessar ou modificar partes do código.



Principais modificadores

Private

Acessível apenas dentro da própria classe.

Padrão

Acessível apenas dentro do mesmo pacote.

Protected

Acessível dentro do mesmo pacote e em subclasses, mesmo que estejam em pacotes diferentes.

Public

Acessível de qualquer lugar.

Herança

Em Java, a herança é uma característica fundamental da programação orientada a objetos, que permite criar classes derivadas com base em classes existentes. A visibilidade dos membros (campos e métodos) de uma classe base em uma classe derivada depende dos modificadores de acesso usados nas duas classes.



Exemplo de modificadores de acesso

java

```
public class Exemplo {  
    private int atributoPrivado;  
    int atributoPadrao;  
    protected int atributoProtegido;  
    public int atributoPublico;  
}
```

Os atributos têm diferentes níveis de visibilidade.

Vamos praticar?

Formem grupos para realizar a atividade.

Primeiro momento

15 minutos

Pesquisem no celular e criem um *card* que mostre um exemplo de *software* que utiliza um dos quatro tipos de modificadores e como ele faz isso.

Segundo momento

10 minutos

Apresentem para a turma o que vocês produziram.



Bloco 2

Modificador *private*

O modificador *private* é um dos quatro principais modificadores de acesso em Java e é usado para restringir o acesso a membros de uma classe.

Por que usar o modificador *private*?

Controle de acesso

O modificador *private* impede que membros da classe sejam acessados de fora da própria classe.

Encapsulamento

Ajuda a alcançar o encapsulamento, ocultando detalhes internos da implementação da classe.

Uso do *private*

java

```
public class MinhaClasse {  
    private int meuAtributo;  
  
    private void meuMetodo() {  
        // Implementação do método  
    }  
}
```

O atributo e o método são declarados como *private*.

Herança

Os métodos privados são visíveis apenas dentro da classe base. Eles não são acessíveis nas classes derivadas.

```
class ClasseBase {  
    private void metodoPrivado() {  
        // código aqui  
    }  
}  
  
class ClasseDerivada extends ClasseBase {  
    // Tentar acessar o método privado aqui resultaria em erro de compilação  
}
```

Herança

Os métodos privados só podem ser acessados dentro da própria classe base e não são visíveis nas classes derivadas. Eles são usados principalmente para encapsular detalhes de implementação da classe base.

A partir disso, em duplas, analisem o exemplo que receberam.



Comportamento do modificador de acesso *private*

Acesso	Mesmo Pacote	Pacotes Diferentes
Classes Derivadas	X	X
Classes não relacionadas	X	X

Vamos praticar?

Formem grupos para realizar a atividade.

Atividade

20 minutos

Criem uma classe com membros privados.



Bloco 3

Modificador padrão

O modificador padrão (ou nenhum modificador) não é uma palavra-chave, mas, sim, a ausência de qualquer modificador explícito. Ele permite que os membros de uma classe sejam acessados dentro do mesmo pacote.



Características do modificador padrão

Pacotes

Membros com o modificador padrão são acessíveis de outras classes no mesmo pacote.

Acesso

Não é possível acessá-los de fora do pacote.

java

```
package meuPacote;

class MinhaClasse {
    int meuAtributo; // Modificador padrão
}
```

Por que usar o modificador padrão?

Controle de acesso

O modificador padrão permite um nível intermediário de acesso, mais restrito que o *public* e menos restrito que o *private*.

Organização

Ajuda a organizar classes e membros em pacotes relacionados.

Exemplo

```
package meuPacote;

public class OutraClasse {
    public static void main(String[] args) {
        MinhaClasse objeto = new MinhaClasse();
        objeto.meuAtributo = 42; // Acesso dentro do mesmo pacote
    }
}
```

A classe "OutraClasse" pode acessar "meuAtributo" de "MinhaClasse" porque ambas estão no mesmo pacote "meuPacote".

Herança

```
package pacote1;

class ClasseBase {
    void metodoPadrao() {
        // código aqui
    }
}

package pacote2;

class ClasseDerivada extends pacote1.ClasseBase {
    // Tentar acessar o método padrão aqui resultaria em erro de compilação, a menos que pacote1 e pacote2
    // estejam no mesmo pacote.
}
```

Herança

Métodos com modificador de acesso padrão são acessíveis apenas dentro do mesmo pacote.

Com base nisso, em duplas, analisem o exemplo que receberam.



Comportamento do modificador de acesso padrão

Acesso	Mesmo Pacote	Pacotes Diferentes
Classes Derivadas	✓	✗
Classes não relacionadas	✓	✗

Vamos praticar?

Formem grupos para realizar a atividade.

Atividade

20 minutos

Criem um pacote e, dentro dele, criem duas classes. Depois, adicionem nele o uso do modificador padrão.



Bloco 4

Modificador *protected*

O modificador "protected" é uma palavra-chave em Java. Ele permite que membros de uma classe sejam acessados dentro do mesmo pacote e em subclasses, independentemente do pacote em que as subclasses estão.



Características do modificador *protected*

Pacotes

Membros com o modificador *protected* são acessíveis por outras classes no mesmo pacote.

Acesso

Membros *protected* também são acessíveis em subclasses, mesmo que estejam em pacotes diferentes.

java

```
package meuPacote;

public class MinhaClasse {
    protected int meuAtributo; // Modificador protected
}
```

Por que usar *protected*?

Facilita a herança

Os membros *protected* podem ser herdados por subclasses, permitindo a extensão de funcionalidades.

Controle de acesso restrito

Fornece um nível intermediário de controle, permitindo acesso apenas a classes relacionadas.

Exemplo

```
java

package outroPacote;

import meuPacote.MinhaClasse;

public class MinhaSubClasse extends MinhaClasse {
    public void modificarAtributo() {
        meuAtributo = 42; // Acesso em uma subclasse
    }
}
```

Neste exemplo, "MinhaSubClasse" pode acessar "meuAtributo" de "MinhaClasse" porque herda a classe "MinhaClasse".

Herança

```
package pacote1;

class ClasseBase {
    protected void metodoProtegido() {
        // código aqui
    }
}

package pacote2;

class ClasseDerivada extends pacote1.ClasseBase {
    // É possível acessar o método protegido aqui, pois ClasseDerivada é uma subclasse de ClasseBase.
}

class OutraClasse {
    // Também é possível acessar o método protegido aqui, pois OutraClasse está no mesmo pacote que ClasseBase.
}
```

Herança

Métodos com modificador de acesso *protected* são acessíveis apenas dentro do mesmo pacote. **Com base nisso, em duplas, analisem o exemplo que receberam.**



Comportamento do modificador de acesso *protected*

Acesso	Mesmo Pacote	Pacotes Diferentes
Classes Derivadas	✓	✓
Classes não relacionadas	✓	✗

Vamos praticar?

Formem grupos para realizar a atividade.

Atividade

20 minutos

Criem uma classe base com membros *protected*.



Bloco 5

Modificador *public*

O modificador *public* é uma palavra-chave em Java. Ele permite que membros de uma classe sejam acessados de qualquer lugar, ou seja, de qualquer classe ou pacote.

Características do modificador *public*

Acesso

Membros com o modificador *public* são acessíveis de qualquer lugar, de qualquer classe ou pacote.

Liberdade

Esse é o nível de acesso mais amplo em Java.

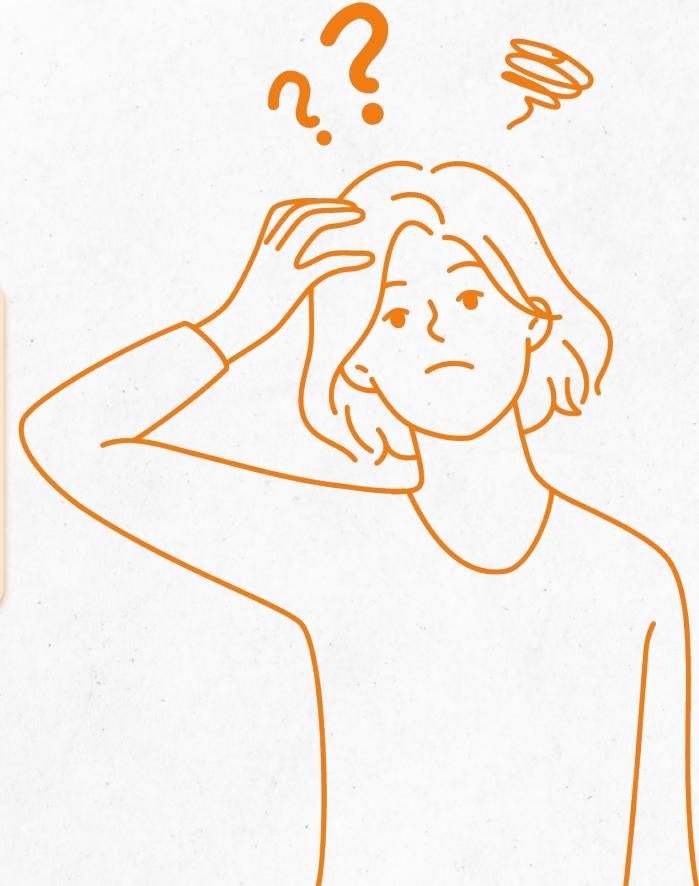
java

```
public class MinhaClasse {  
    public int meuAtributo; // Modificador public  
}
```

Quando usar?

O modificador *public* é apropriado quando você deseja que um membro seja acessível de forma ampla, sem restrições.

É comum usá-lo para métodos ou atributos que são parte da interface pública de uma classe.



Exemplo

```
java

public class Calculadora {
    public int somar(int a, int b) {
        return a + b;
    }
}
```

Neste exemplo, o método "somar" é público, o que significa que pode ser chamado de qualquer classe ou pacote.

Herança

```
class ClasseBase {  
    public void metodoPublico() {  
        // código aqui  
    }  
}  
  
class ClasseDerivada extends ClasseBase {  
    // É possível acessar o método público aqui.  
}  
  
class OutraClasse {  
    // Também é possível acessar o método público aqui.  
}
```

Herança

Métodos públicos são visíveis em qualquer lugar, tanto dentro quanto fora da classe base e suas classes derivadas, independentemente do pacote.

Com base nisso, em duplas, analisem o exemplo que receberam.



Comportamento do modificador de acesso *public*

Acesso	Mesmo Pacote	Pacotes Diferentes
Classes Derivadas	✓	✓
Classes não relacionadas	✓	✓

Vamos praticar?

Formem grupos para realizar a atividade.

Atividade

20 minutos

Criem uma classe base com membros *public*.



Bloco 6

O que vem à sua mente quando ouve falar
sobre os tipos de modificadores de acesso?



Vamos praticar?

Formem grupos para realizar a atividade.

Primeiro momento

25 minutos

Criem um código utilizando um dos quatro tipos de modificadores de acesso.

Segundo momento

5 minutos

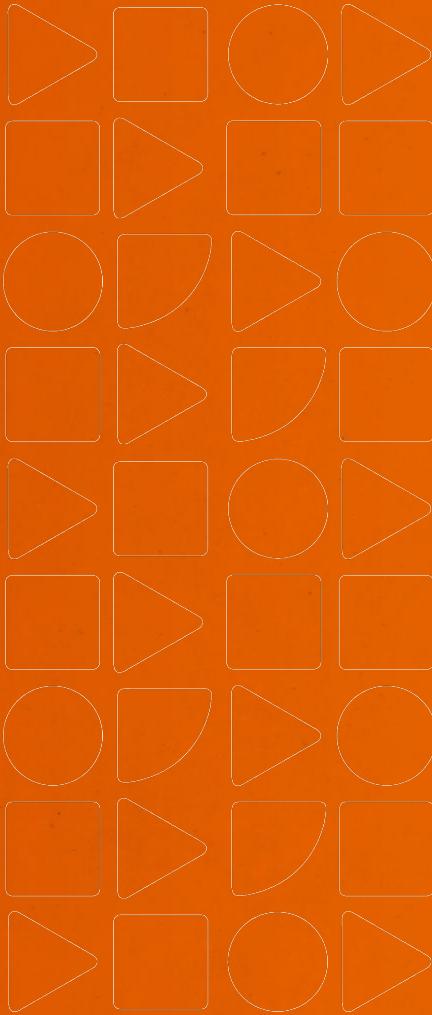
Falem para a turma sobre as facilidades e dificuldades ao construir o código.



Fechamento

Diga uma palavra sobre o que você achou da aula.





Referências Bibliográficas

PROZ EDUCAÇÃO. *Apostila de Desenvolvimento de Aplicações*. 2023