

### Métodos Abstratos







#### **Temas**

Classes Abstratas em Java

Sobreescrevendo

**Métodos Abstratos** 

Métodos Abstratos em Java

Atributos e Métodos em **Classes Abstratas** 

Métodos Abstratos

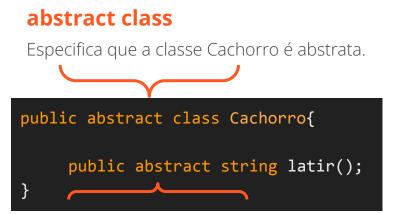


## 1 Classes Abstratas em Java



#### Classes Abstratas em Java

Definimos classes abstratas e métodos abstratos com a palavra-chave "abstract". Como o método é abstrato (só dizemos o que fazer), eles não possuem código associado, não possuem "corpo". Vejamos um exemplo:



Especifica que o método latir() é abstrato.



## 2 Métodos Abstratos em Java



#### Métodos Abstratos em Java

**Se o Doberman quiser SER UM CACHORRO**, ele deve respeitar o contrato dos Cachorros: ele deve implementar um método chamado latir, que retorna uma String e não recebe parâmetros. Simplificando, você deve **sobrescrever todos os métodos abstratos definidos em Cachorro**.

```
public class Doberman extends Cachorro{

   public string latir() {
      return "Tenho um latido de um
doberman AUAU!!!";
   }
}
```





#### Métodos Abstratos em Java

Ou seja, se Cachorro diz que todos os cães devem latir(), porém o Doberman deve "explicar" como fazê-lo. Chamaremos essa operação de "implementar" o método latir().

public class Doberman extends Cachorro{
}



Neste caso, a classe Doberman irá lançar um erro de compilação, pois não respeita o contrato da classe Cachorro.



## 3 Sobrescrevendo Métodos abstratos



#### Sobreescrevendo métodos abstractos

Se Poodle quiser SER um cachorro, ele também deve sobrescrever todos os métodos abstratos definidos em Cachorro.

```
public class Poodle extends Cachorro{
    public string latir() {
        return "Tenho um latido de um Poodle;
    }
}
```



Quando implementamos os métodos, eles não são mais abstratos, portanto, tanto no Doberman quanto no Poodle não usamos mais a palavra-chave abstract.







As regras para implementar métodos abstratos são sobrescrever, portanto as mesmas regras se aplicam: respeitar tipo, quantidade e ordem dos parâmetros.



Se não fizermos isso, então não respeitaremos o contrato, se não respeitarmos o contrato, a classe vai lançar um erro de compilação.



# Atributos e Métodos em Classes Abstratas



#### Atributos e Métodos em Classes Abstratas

Uma classe abstrata é uma classe como qualquer outra, portanto, pode ter atributos e métodos concretos. Mesmo assim, devemos lembrar que apenas os abstratos serão os que definem o contrato.

```
public abstract class Cachorro{
    private String nome;
     public void setNome(String nome){
          this.nome = nome;
     public String getNome(){
          return nome;
     public abstract string latir();
```



#### Atributos e Métodos em Classes Abstratas

Por que ter métodos concretos em um caso que não pode ser instanciado? Porque esses métodos podem ser reutilizados.

Por outro lado, o fato de uma classe abstrata não poder ser instanciada não significa que ela não possa ter construtores, o objetivo é o mesmo: podemos definir construtores para reutilizar código quando herdamos dessa classe abstrata.







#### **Exemplo:**

```
public class Principal{
     public static void main(String[] args){
          Doberman cachorro1 = new Doberman();
                   cachorro1.latir();
          Poodle cachorro2 = new Poodle();
                  cachorro2.latir();
```



#### Exemplo:

```
public class Principal{
     public static void main(String[] args){
          Cachorro cachorro1 = new Cachorro();
                   cachorro1.latir();
                                                             Isso falhará ao
                                                             compilar. Lembre-se
                                                             de que, por definição,
                                                             uma classe abstrata
                                                             NÃO PODE ser
                                                             instanciada.
```

## DigitalHouse>