

Aula 04

- 1. Palavras Reservadas
- 2. Criação de Objectos
- 3. Métodos Construtores
- 4. Variáveis de Instância
- 5. Strings
- 6. Palavras Reservadas Usadas
- 7. Links Úteis



abstract assert*** boolean break byte case catch char class const*

continue for default do double else enum**** extends final finally float

not used

added in 1.2

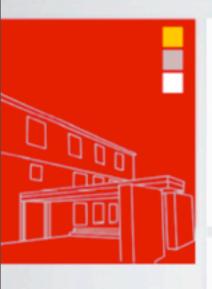
added in 1.4

added in 5.0

goto* if implements import instanceof return int interface long native

new package private protected public short static strictfp** super

switch synchronized throw throws transient try void volatile while



Objecto

- Um objecto é uma instância de uma classe.
- Construído a partir da especificação de uma classe.
- Com uma identidade única.
- Em java para criarmos um objecto usamos o operador de alocação new como podemos ver no slide seguinte.

Os objectos são alocados dinamicamente e o operador de alocação é o new:

Sintaxe de alocação



```
public class Dog {
   char[] name = {'f', 'i', 'd', 'o'};
   char[] bark = {'w', 'o', 'o', 'f','!'};
   int age = 6;
}
```

Os atributos name, bark e age da classe Dog são denominados variáveis de instância.

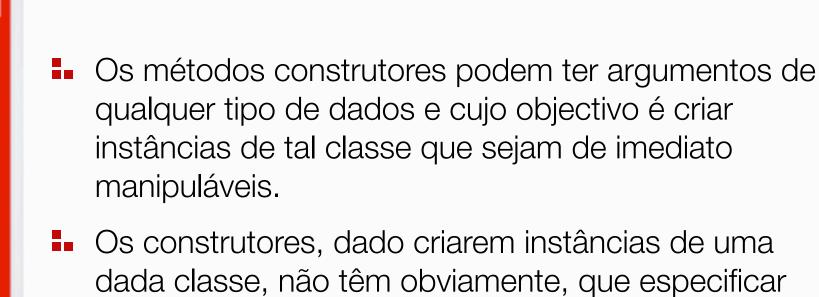
```
class ADogsLife {
  public static void main(String[] args) {
     Dog fido = new Dog();
                                       new é o operador
                                       de alocação.
     System.out.println(fido.name);
     System.out.println(fido.bark);
                                       Dog() é o método
     System.out.println(fido.age);
                                       constructor.
 Reparou que não existe o membro "Dog ()"
```

```
class ADogsLife {
  public static void main(String[] args) {
    Dog fido = new Dog();
    Dog spot = new Dog();
    System.out.println(fido.name);
    System.out.println(fido.bark);
    System.out.println(fido.age);
    System.out.println(spot.name);
    System.out.println(spot.bark);
    System.out.println(spot.age);
```



Métodos Construtores

Os construtores de uma classe são todos os métodos especiais que são declarados na classe tendo por identificador o nome exacto da classe.



- Os construtores, dado criarem instâncias de uma dada classe, não têm obviamente, que especificar qual o resultado, pois será sempre uma instância da respectiva classe.
- É possível e útil construir mais do que um construtor de instâncias de uma dada classe.

```
public class Dog {
    char[] name = {'f', 'i', 'd', 'o'};
    char[] bark = {'w', 'o', 'o', 'f', '!'};
    int age = 6;

    Dog(char[] nametmp)
    {
        name = nametmp;
    }
}
```

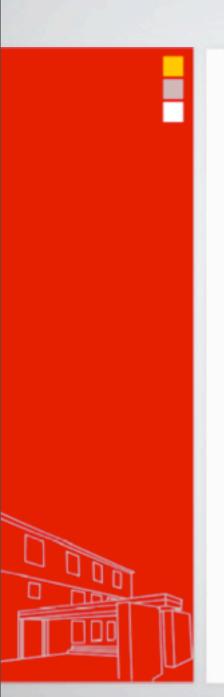
Dog(char[] nametmp) é o método construtor

```
class ADogsLife {
  public static void main(String[] args) {
    char[] fidoname = {'f', 'i', 'd', 'o'};
    Dog fido = new Dog(fidoname);
    char[] spotname = {'s', 'p', 'o', 't'};
    Dog spot = new Dog(spotname);
    System.out.println(fido.name);
    System.out.println(fido.bark);
    System.out.println(fido.age);
    System.out.println(spot.name);
    System.out.println(spot.bark);
    System.out.println(spot.age);
```

```
public class Dog {
  char[] name = {'f', 'i', 'd', 'o'};
  int age = 6;
  Dog(char[] nametmp)
    name = nametmp;
    name = nametmp;
    bark = barktmp;
     age = agetmp;
Criámos um novo método construtor.
```

char[] bark = {'w', 'o', 'o', 'f', '!'}; Dog(char[] nametmp, char[] barktmp, int agetmp)

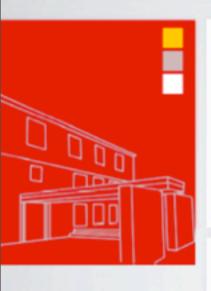
```
class ADogsLife {
  public static void main(String[] args) {
    char[] dogname={'s', 'p', 'o', 't'};
    char[] dogbark={'r', 'u', 'f', 'f', '!'};
    Dog spot = new Dog(dogname, dogbark,3);
    System.out.println(spot.name);
    System.out.println(spot.bark);
    System.out.println(spot.age);
```



Variáveis de Instância

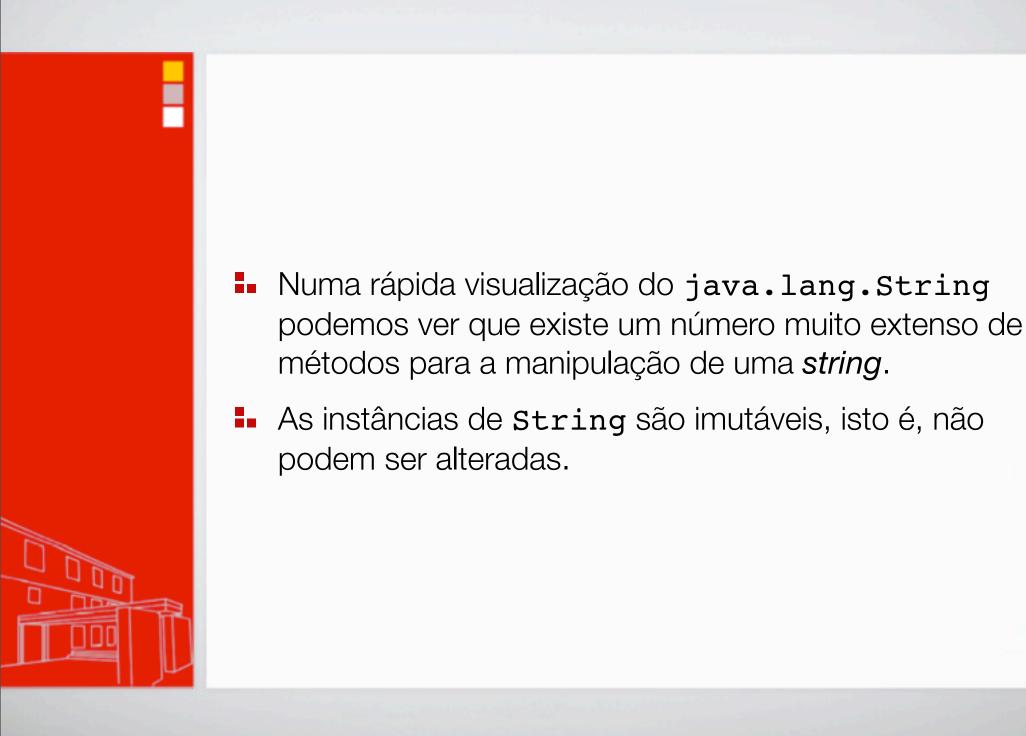
- Até agora aprendemos a definir instâncias de objectos mas cada objecto pode também ter variáveis de estado.
- Para cada instância de **Dog** do exemplo anterior podemos ter alguma variabilidade de objectos de acordo com as suas características tais como cor do pêlo, peso, etc.
- A estas variáveis damos o nome de variáveis de instância.

```
public class Dog {
  char[] name;
 char[] bark = {'w', 'o', 'o', 'f', '!'};
  int age = 6;
                             Variáveis de Instância
  Dog(char[] nametmp)
    name = nametmp;
  Dog(char[] nametmp, char[] barktmp, int agetmp)
    name = nametmp;
    bark = barktmp;
    age = agetmp;
                              Métodos Construtores.
```



String

- A String é um tipo não primitivo que está definido numa classe de sistema do Java java.lang
- O package lang é considerado tão essencial que não é necessário indicar o caminho das classes que pretendemos utilizar deste package.



Uma normal operação de display envolve strings:

```
System.out.println("x= " + x);
```

Como o "+" é reconhecido pelo compilador de Java como um operador de concatenação de *strings*, o compilador automaticamente gera o código necessário para converter qualquer operando que não seja uma *String* numa instância de *String*.

Como String é uma classe a forma geral para criar a instância de uma string é a seguinte:

```
String prompt = new String("x= ");
```

- De modo a ser mais cómodo para o programador a linguagem Java reconhece uma sequência de caracteres entre aspas como uma constante String
- Podemos então criar uma instância de **String** de uma forma mais rápida:

```
String prompt = "x= ";
String barksound = "woof!";
```

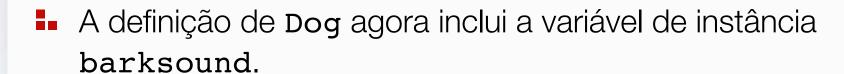
- Vamos alterar a nossa Classe Dog de forma a usar uma String barksound em vez do array de char bark.
- Alterar esta linha:

```
char[] bark = {'w','o','o','f','!'};
```

Por esta:

```
String barksound = "woof!";
```

```
public class Dog {
  char[] name;
  String barksound = "woof!";
  int age = 6;
                             Variáveis de Instância
  Dog(char[] nametmp)
    name = nametmp;
  Dog(char[] nametmp, String barktmp, int agetmp)
    name = nametmp;
    barksound =barktmp;
     age = agetmp;
                              Métodos Construtores.
```



- Cada vez que é criada uma nova instância de **Dog** vai incluir uma referência para a instância da **String** que representa o ladrar do cão.
- A linha

```
String barksound = "woof!";
```

aloca a instância de uma **String** e inicializa-a com o valor de "Woof!".

De seguida podemos alterar também a variável de instância name: char[] name; por: String name;

```
public class Dog {
  String name;
  String barksound = "woof!";
  int age = 6;
                             Variáveis de Instância
  Dog(String nametmp)
    name=nametmp;
  Dog(String nametmp, String barktmp, int agetmp)
    name = nametmp;
    barksound = barktmp;
     age = agetmp;
                              Métodos Construtores.
```

Palavras reservadas usadas

abstract assert*** boolean break byte case catch char class const*

continue for default do double else enum**** extends final finally float

not used

added in 1.2

added in 1.4

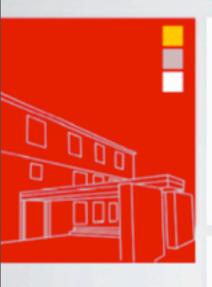
added in 5.0

goto* if implements import instanceof return int interface long native

new package private protected public short static strictfp** super

switch synchronized this throw throws transient try void volatile while





Links Úteis

- http://docs.oracle.com/javase/tutorial/java/javaOO/ objects.html
- http://docs.oracle.com/javase/tutorial/java/javaOO/ objectcreation.html
- http://docs.oracle.com/javase/tutorial/java/javaOO/ variables.html
- http://docs.oracle.com/javase/tutorial/java/data/strings.html