

Programação Orientada a Objetos

Profa Andréa Schwertner Charão
DLSC/CT/UFSM

Introdução a Java

- Linguagem semelhante a C/C++
 - mais recente e mais simples
- Portabilidade (“*write once, run anywhere*”)
 - código objeto intermediário (*bytecode*) interpretado por máquinas virtuais (JVMs)
- Amplo conjunto de APIs (classes reutilizáveis)
 - p.ex.: programação para Web, interfaces gráficas, etc.
- Suporte a programação concorrente

Java versus C/C++

- Estruturas de repetição, seleção e seqüência são praticamente idênticas
- Tipos de dados **primitivos** são basicamente os mesmos (short, int, long, float, double, char), com algumas diferenças:
 - não há números unsigned
 - os tamanhos são sempre os mesmos em qualquer arquitetura
 - tipos boolean e byte
- Arrays e strings são **tipos abstratos de dados**

Java versus C/C++

- Expressões aritméticas/relacionais são formadas como em C/C++
- Operadores aritméticos/relacionais são praticamente idênticos
- Não há ponteiros em Java! (mas há referências)
- Sintaxe dos comentários é idêntica

Tipos abstratos e encapsulamento em Java

- Suporte ao encapsulamento/TAD: **class**
- Classes
 - “Parente” do struct em C
 - definição de **atributos/campos**
 - definição de **métodos**

```
class Data
{
    atributos {
        private int dia;
        private int mes;
        private int ano;
    }
    métodos {
        public void mostra() { ... }
    }
}
```

Tipos abstratos e encapsulamento em Java

- Suporte ao encapsulamento/TAD: **class**
- Suporte à ocultação/visibilidade
 - cláusula **private**: atributos/métodos visíveis somente dentro da classe
 - cláusula **public**: atributos/métodos visíveis fora da classe

```
class Data
{
    private int dia;
    private int mes;
    private int ano;

    public void mostra() { ... }
}
```

atributos {

métodos {

tipo de visibilidade

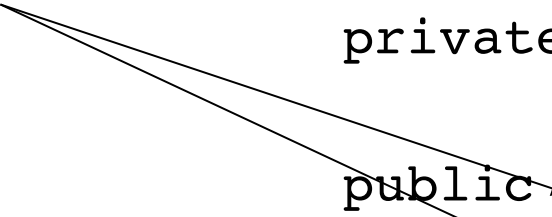
Classes em Java

■ Construtores em Java

construtores

```
class Data
{
    private int dia;
    private int mes;
    private int ano;

    public Data() { ... }
    public Data(int dia, int mes, int ano)
    { ... }
    public void mostra() { ... }
}
```



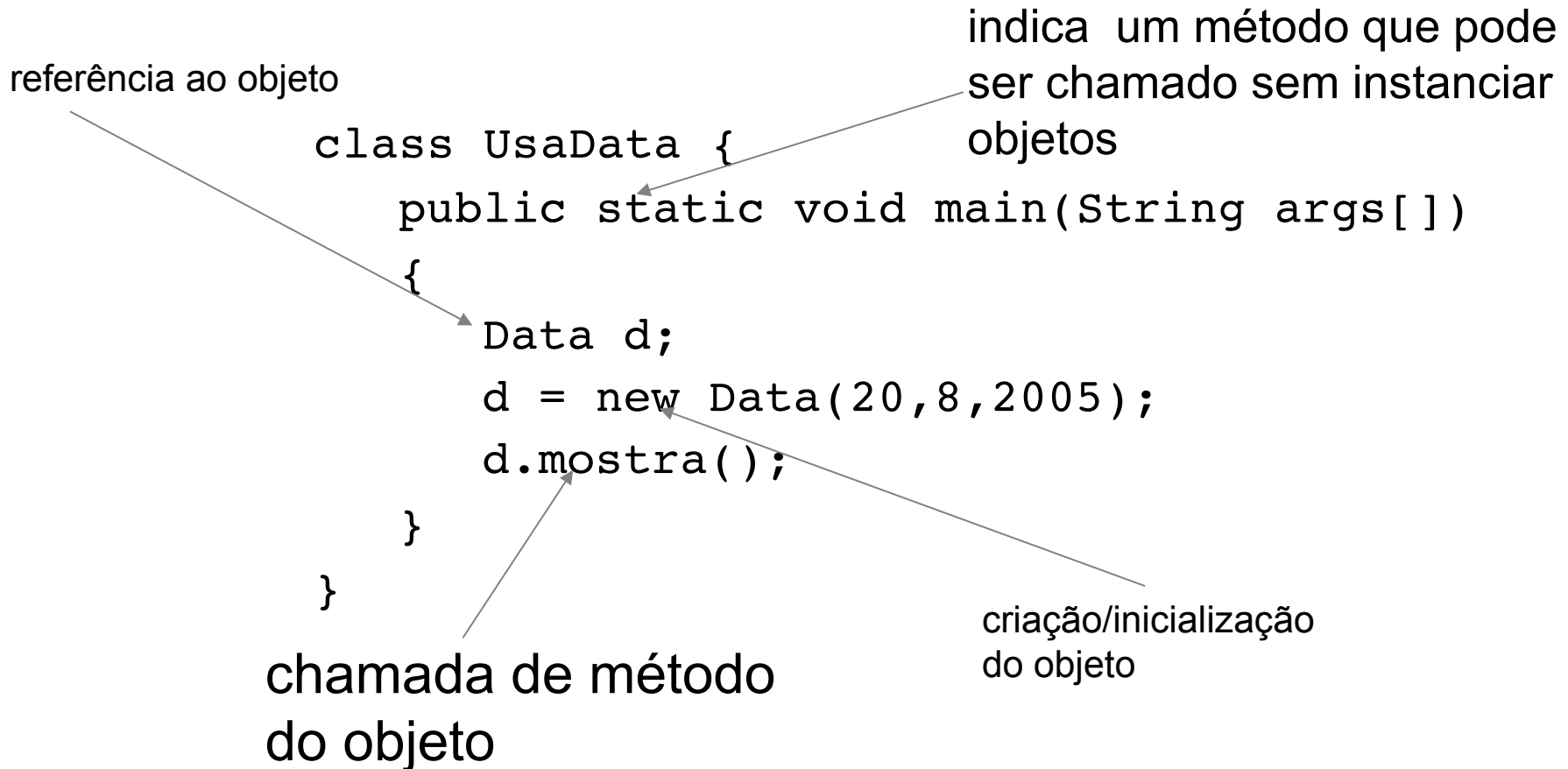
Classes em Java

■ Construtores

- servem para inicialização de objetos
- possuem o mesmo nome da classe
- não possuem tipo de retorno
- uma classe pode ter tantos construtores quantos forem necessários
- a diferenciação entre os construtores se dá pelos seus argumentos (tipo e número)

Código usuário da classe

■ Objetos e referências a objetos



Em Java...

- Ao contrário de C++, tudo deve estar dentro de classes (incluindo a função principal do programa)
- Não há funções ou dados globais (só static)
- Todas as definições de métodos são feitas no corpo da classe (não ficam num .h)
- Não há **destrutores** (o programador não se preocupa em desalocar objetos na memória)
- **Coletor de lixo** desaloca memória para objetos que não são mais referenciados

Programa "Olá, mundo"

```
class HelloApp
{
    public static void main(String[] args)
    {
        System.out.println("Hello World!");
    }
}
```

classe java.lang.System

método public da classe
PrintStream

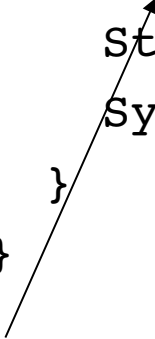
atributo public static PrintStream

Veja: <http://docs.oracle.com/javase/tutorial/getStarted/application/index.html>

Programa "Olá, usuário"

```
class HelloUser
{
    public static void main(String[] args)
    {
        System.out.println("Qual seu nome?");
        Scanner input = new Scanner(System.in);
        String nome = input.next();
        System.out.println("Olá, " + nome + "!");
    }
}
```

Classe Scanner
métodos para int,
float, string etc



atributo public static InputStream



Scanner

```
6 package hello;
7
8 import java.util.Scanner;
9
10 /**
11  *
12  * @author cassales
13  */
14 public class Hello {
15
16     /**
17      * @param args the command line arguments
18      */
19     public static void main(String[] args) {
20         System.out.println("Hello, World!");
21         Scanner input = new Scanner(System.in);
22         String nome = input.next();
23
24     }
25
26 }
27
```

[java.util.Scanner](#)

public [String](#) next()

Finds and returns the next complete token from this scanner. A complete token is preceded and followed by input that matches the delimiter pattern. This method may block while waiting for input to scan, even if a previous invocation of [Scanner.hasNext](#) returned true.

Retorna:

the next token

Gera:

[NoSuchElementException](#) - if no more tokens are

next()	String
next(Pattern pattern)	String
next(String pattern)	String
nextLine()	String
nextBigDecimal()	BigDecimal
nextBigInteger()	BigInteger
nextBigInteger(int radix)	BigInteger
nextBoolean()	boolean
nextByte()	byte
nextByte(int radix)	byte
nextDouble()	double
nextFloat()	float
nextInt()	int
nextInt(int radix)	int
nextLong()	long
nextLong(int radix)	long
nextShort()	short

System

OVERVIEW PACKAGE **CLASS** USE TREE DEPRECATED INDEX HELP Java™ Platform
Standard Ed. 8

PREV CLASS NEXT CLASS FRAMES NO FRAMES ALL CLASSES

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

compact1, compact2, compact3
java.lang

Class System

java.lang.Object
 java.lang.System

public final class **System**
extends Object

The System class contains several useful class fields and methods. It cannot be instantiated.

Among the facilities provided by the System class are standard input, standard output, and error output streams; access to externally defined properties and environment variables; a means of loading files and libraries; and a utility method for quickly copying a portion of an array.

Since:
JDK1.0

Field Summary

Fields

Modifier and Type	Field and Description
static <code>PrintStream</code>	err The "standard" error output stream.
static <code>InputStream</code>	in The "standard" input stream.
static <code>PrintStream</code>	out The "standard" output stream.

Method Summary

All Methods Static Methods Concrete Methods Deprecated Methods

Modifier and Type	Method and Description
static void	arraycopy (<code>Object</code> src, int srcPos, <code>Object</code> dest, int destPos, int length) Copies an array from the specified source array, beginning at the specified position, to the specified position of the destination array.

<http://docs.oracle.com/javase/8/docs/api/java/lang/System.html>

Plataforma Java

- Java SE: Standard Edition
 - JDK (desenvolvimento, inclui JRE)
 - JRE (só p/execução)
- Java EE: Enterprise Edition
- Java ME: Micro Edition

Plataforma Java: execução

