Noções básicas da linguagem Java

Prof. Marcel Hugo

Departamento de Sistemas e Computação
Universidade Regional de Blumenau - FURB

1

Uso de Orientação a Objetos em Java

Objetos e Mensagens
Objetos em Java
Tipos primitivos
Classes básicas (*wrappers*)
String
Conversão de tipos
Containers (Matrizes)

2

Um pouco de história...

- Variáveis como elementos isolados
  - int dia, mes, ano;
- Fácil compreensão, porém duas principais desvantagens:
  - Várias declarações:
    - int meuDiaNascimento, meuMesNascimento, meuAnoNascimento;
    - int seuDiaNascimento, seuMesNascimento, seuAnoNascimento;
  - Variáveis independentes, sem relação:
    - Em nível conceitual, tudo faz parte de uma data.
- Para isto, há os tipos agregados de dados (estruturados ou de registro).

3

Tipos agregados de dados

- Exemplo Pascal:

```
Data = record
    dia, mes, ano: integer;
end;
```

em C:

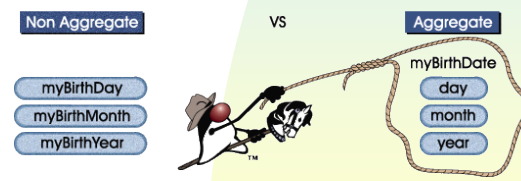
```
struct Data {
    int dia;
    int mes;
    int ano;
}
```

em Java:

```
public class Data {
    int dia;
    int mes;
    int ano;
}
Data meuNascimento, seuNascimento;
```

- Mais tarde, usa-se:

```
seuNascimento.dia = 26;
seuNascimento.mes = 11;
seuNascimento.ano = 1960;
```



- Porém, ainda há que se definir métodos, pois como está qualquer valor inteiro pode ser atribuído às variáveis membro (atributos).

4

Tipos abstratos de dados (TAD)

- Sobre uma estrutura de dados pode-se definir e executar um conjunto específico de operações.
- O TAD encapsula a estrutura de dados. Os usuários do TAD só tem acesso a algumas operações disponibilizadas sobre esses dados.
- Usuário do TAD x Programador do TAD

Usuário só “enxerga” a interface, não a implementação:

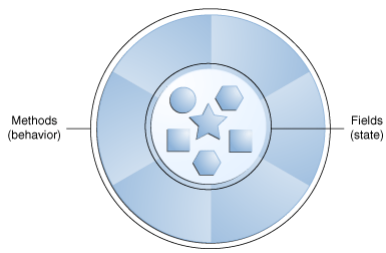
- Dessa forma, o usuário pode abstrair da implementação específica.
- Qualquer modificação nessa implementação fica restrita ao TAD/

5

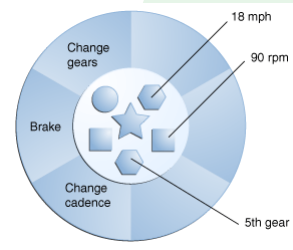
Objetos e Classes

- Classe com seus atributos (Data com dia, mês e ano) e suas operações.
- Objeto é uma instância da classe (meuNascimento)

Classe



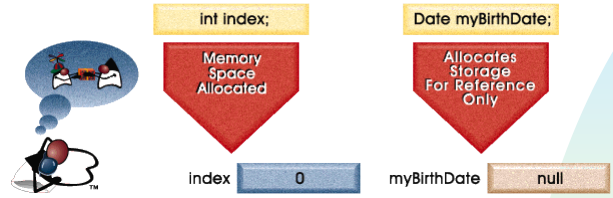
Objeto



6

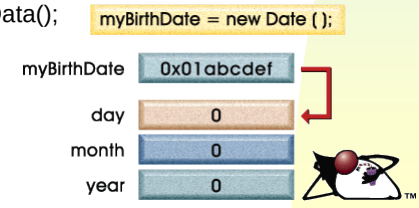
Criação de um objeto

- Declaração não cria objeto, diferentemente dos tipos primitivos



- Para criar efetivamente, deve-se instanciá-lo.

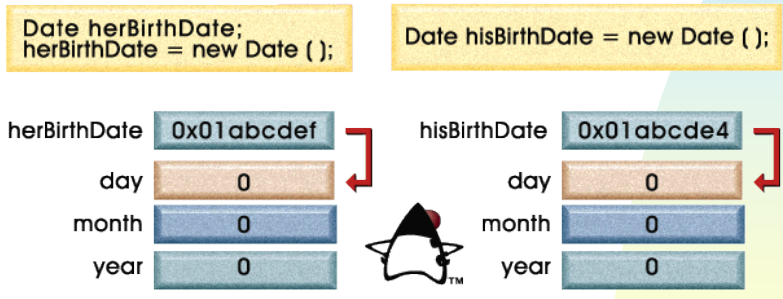
```
Data meuNascimento;
meuNascimento = new Data();
```



7

Criação de um objeto (continuação)

- O espaço real de memória alocado com ***new*** é o objeto efetivamente.



8

Características dos Objetos

- Os objetos não são acessados diretamente
  - são acessados por meio de um manipulador (*handle*), através do qual lhe são enviadas mensagens.
  - Ex.: lulu = new Cachorro();
    // lulu é o *handle* do novo objeto
- Os objetos existem à parte do fluxo do programa
  - estão todos na *heap* (alocados dinamicamente)
  - não tem tempo de vida determinado pelo programador
  - não tem escopo, na verdade, são todos visíveis a partir de um manipulador que tem escopo e tempo de vida
  - quando não existem mais referências para um objeto ele é automaticamente retirado da memória (*garbage collection*)

9

Utilização de Objetos em Java

- Declaração do *handle*
  - *ClasseTipo nomeHandle*;
  - Ex.:
    - Date hoje;
- Instanciação
  - Associação do *handle* a um novo objeto
    - *nomeHandle* = **new** *ConstrutorDaClasse*();
  - Associação do handle a um objeto que já existe
    - *nomeHandle* = *outroHandle*;

10

Exemplo de uso de objetos

```
...
Date hoje;
hoje = new Date();  // cria uma instância da classe Data
          // a data de hoje,
          // sendo 'hoje' o handle deste objeto

Date aniv = new Date(2007, 5, 30);  // construtor hipotético
Date aniv2;
aniv2 = aniv;    // aniv e aniv2 são handles para o
          // mesmo objeto
if (hoje.after(aniv) { // já passou do aniv? Hoje depois de aniv?
   // envio da mensagem after para o objeto hoje.
   // parâmetro é o objeto aniv
}
```

11

Tipos primitivos e seus valores

- Inteiros
  - **byte**: 8 bits, -128 a 127
  - **short**: 16 bits, -32768 a 32767
  - **int**: 32 bits, -2147483648 a 2147483647
  - **long**: 64 bits, ... (200L - literal 200 long)
- Reais
  - **float**: 32 bits (1f - literal 1 float; 1e+9f)
  - **double**: 64 bits (1d - literal 1 double; 47e-341d)
- Caracter
  - **char** (Unicode caracter): 16 bits ('a' - literal)
- Lógicos
  - **boolean** (1 bit). Valores literais: { true, false }

12

Literais

- Literais inteiros (*int*)
  - Decimais: 1 , 2, 45 ,...
  - Octais: 07, 010
  - Hexadecimais: 0xff (255)
- Literais de ponto flutuante – decimais com fração (*double*)
  - 2.0 , 3.14159 , ...
  - 314159e-05 , 314159E-05
- Literais booleanos (*boolean*)
  - *true* e *false*
- Literais de caracteres (*char*)
  - Valores de 16 bits que podem ser manipulados como inteiros
  - Par de apóstrofos (‘ ‘) ‘a’
  - Seqüência de escape (\ ) ‘ \141’ octal = ‘ \u0061’ hex = ‘a’

13

Classes Básicas x Tipos Primitivos

- Variáveis (par: posição memória + valor)

```
int a = 10;        // a é um tipo primitivo,
                   // não pode receber mensagens, ex.: a.inc()
```

- Classes básicas (*wrappers*) e seus objetos

```
// b é um handle de objeto
Integer b = new Integer(10);
int c = b.intValue();
```

- Tipos primitivos ficam na *stack* (mundo do programa)
- Objetos ficam na *heap* (mundo dos objetos)

Tipo primitivo	Classe Wrapper
boolean	Boolean
byte	Byte
char	Character
double	Double
float	Float
int	Integer
long	Long
short	Short

14

A classe String

- Representa qualquer seqüência de caracteres
- Valores: “exemplo de valor literal string”

  String nome = “João da Silva”;

  nome.toUpperCase(); // mensagem para objeto nome.
- Operador:

  + (concatenação)

  nome = nome + “sauro”;
- Observações
  - Strings são objetos, não são tipos primitivos, porém Java oferece facilidades de manipulação.
  - Strings são objetos que não mudam de valor

15

Principais métodos da classe String

- boolean equals(String s)
  - retorna true se a string é igual a ***s***
- boolean equalsIgnoreCase(String s)
  - retorna true se a string é igual a ***s*** independente de maiúsculo/minúsculo
- int length()
  - retorna o tamanho da string
- int indexOf(String s)
  - procura a ***s*** na string e retorna a posição, retorna -1 se não achou
- char charAt(int i)
  - retorna o caracter na posição ***i*** da string (começa de 0)

16

Conversão de tipos

- Tipos primitivos (**cast**)
  - char a;
  - int i = (int)a;
  - int x = (int)10L;
- String em número
  - String piStr = "3.14159";
  - Float pi1 = Float.valueOf(piStr);
  - float pi2 = Float.parseFloat(piStr);
  - String exemplo = “35”;
  - int quantidade1 = Integer.parseInt(exemplo);
  - Integer quantidade2 = Integer.valueOf(exemplo);
  - Integer quantidade3 = new Integer(exemplo);

17

Entendendo o Exemplo1

Tudo em Java funciona no interior de classes.
A estrutura de um programa Java é uma definição de classe.

```
class Exemplo1 {

   public void demo() {
    for (int i=0; i<100; i++) {
        if ((i % 2) == 0)  {
            System.out.println(i);
        }
    }
   }
}
```

18

Declaração de Classes (simplificada)

- Declaração
  class *NomeClasse*
  {
  atributos
  métodos
  }
- Atributos: [*acesso*] *tipo nomeAtributo* [= *expressão*];
- Métodos: [*acesso*] *tipoRetorno nomeMétodo* ([*parâmetros formais*])
  { corpo
  }

19

Antropormofizar....

- Para facilitar, sempre pense no objeto como sendo uma pessoa que então possui informações e age a partir destas informações e com elas.
- As pessoas conversam entre si (trocam mensagens)

20