Programação Orientada à Objetos (POO)

CIÊNCIA DA COMPUTAÇÃO

Prof. Dr. João Paulo Aramuni

UNIVERSIDADE FUMEC

Sumário

* **Identificadores e Tipos**
  * Tipos primitivos
  * Tipos referência
  * Construção e inicialização
  * Referência this

Identificadores e Tipos

- **Tipos primitivos**
  - Há **oito** tipos primitivos (pré-definidos):
    - Números Inteiros
      - **byte** 1 byte
      - **short** 2 bytes
      - **int** 4 bytes
        - mais usado
      - **long** 8 bytes
    - Números em Ponto Flutuante
      - **float** 4 bytes
      - **double** 8 bytes
        - duas vezes a precisão do tipo float
        - mais usado

Identificadores e Tipos

* **Tipos primitivos**

Tipo	Tamanho em *bits*	Faixa
byte	8	-128 até +127
short	16	-32,768 até +32,767
int	32	-2,147,483,648 até+2,147,483,647
long	64	-9,223,372,036,854,775,808 até +9,223,372,036,854,775,807

Tipo	Tamanho em *bits*	Faixa
float	32	-3.40292347E+38 até +3.40292347E+38
double	64	-1.79769313486231570E+308 até +1.79769313486231570E+308

Identificadores e Tipos

* **Tipos primitivos**
  * Há **oito** tipos primitivos (pré-definidos):
    * Caractere
      * **char** 2 byte
        * caracteres Unicode
          * permite até 65536 caracteres (atualmente usados cerca de 35000)
          * primeiros 255 caracteres idênticos ao código ASCII / ANSI.
        * representado por aspas simples. Ex.: “H”.
          * “H” representa uma string contendo um único caractere.
      * **caracteres especiais:**
        * **\b** (backspace)
        * **\t** (tab)
        * **\n** (linefeed)
        * **\r** (carriage return)
        * \” (aspas duplas)
        * \’ (apóstrofe)
        * \\ (barra invertida)

Identificadores e Tipos

* **Tipos primitivos**
  * Há **oito** tipos primitivos (pré-definidos):
    * **Lógico**
      * **boolean** 1 byte (valores verdadeiros (**true**) e falso (**false**))
    * O restante são objetos (exceto array)
    * Sendo assim, os tipos primitivos são **boolean, byte, char, short, int, long, float** e **double**.

Identificadores e Tipos

* **Tipos primitivos**
  * Uma variável do tipo primitivo pode armazenar exatamente um valor de seu tipo declarado por vez, quando outro valor for atribuído a essa variável, seu valor inicial será substituído.
  * As variáveis de instância de tipo primitivo são inicializadas por padrão, as variáveis dos tipos byte, char, short, int, long, float e double são inicializadas com 0, e as variáveis do tipo boolean são inicializadas como **false**.
  * Esses tipos podem especificar seu próprio valor inicial para uma variável do tipo primitivo atribuindo à variável um valor na sua declaração.

Identificadores e Tipos

* **Tipos referência**
  * Os tipos por referência, são classes que especificam os tipos de objeto **Strings, Arrays Primitivos e Objetos**.
  * Os programas utilizam as variáveis de tipos por referência para armazenar as localizações de objetos na memória do computador.
  * Esses objetos que são referenciados podem conter várias variáveis de instância e métodos dentro do objeto apontado.
  * Para trazer em um objeto os seus métodos de instância, é preciso ter referência a algum objeto. As variáveis de referência são inicializadas com o valor "**null**" (nulo).

Identificadores e Tipos

* **Construção e inicialização**
  * A seguir, exemplo de construção e inicialização de tipos primitivos:

Identificadores e Tipos

Construção e inicialização

```
public class Tipos_Primitivos {
    public static void main(String[] args) {
        byte tipoByte = 127;
        short tipoShort = 32767;
        char tipoChar = 'C';
        float tipoFloat = 2.6f;
        double tipoDouble = 3.59;
        int tipoInt = 2147483647;
        long tipoLong = 92233720368547758O7L;
        boolean tipoBooleano = true;
        System.out.println("Valor do tipoByte = " + tipoByte);
        System.out.println("Valor do tipoShort = " + tipoShort);
        System.out.println("Valor do tipoChar = " + tipoChar);
        System.out.println("Valor do tipoFloat = " + tipoFloat);
        System.out.println("Valor do tipoDouble = " + tipoDouble);
        System.out.println("Valor do tipoInt = " + tipoInt);
        System.out.println("Valor do tipoLong = " + tipoLong);
        System.out.println("Valor do tipoBooleano = " + tipoBooleano);
    }
}
```

Identificadores e Tipos

* **Construção e inicialização**
  * A seguir, exemplo de construção e inicialização de tipos referência.
  * A linha: ClasseConta acao = **new** ClasseConta(), cria um objeto de classe ClasseConta e a variável acao contém uma referência a esse objeto ClasseConta, onde poderá invocar todos os seus métodos e atributos da classe.
  * A palavra chave **new** solicita a memória do sistema para armazenar um objeto e inicializa o objeto.

Identificadores e Tipos

* **Construção e inicialização**

```
public class AcessaMetodo {
    public void imprime(){
        System.out.println("Bem Vindo ao Java!");
    }
    public static void main(String[] args) {
        AcessaMetodo acessa =  new AcessaMetodo();
        acessa.imprime();
    }
}
```

* A saída do código acima irá ser reproduzida através da ação acessa.imprime(), porque está sendo acessado o método do objeto que foi inicializado com a variável definida como "acessa".

Identificadores e Tipos

* **Tipos primitivos e tipos referência**
  * Observações:
    * As variáveis de tipos por valor não referenciam objetos, esses tipos de variáveis não podem ser utilizadas para invocar métodos.
    * Lembrar de que as variáveis locais não são inicializadas por padrão (variáveis dentro dos métodos).
    * As variáveis do tipo primitivo não podem ser inicializadas como referência a um objeto.

Identificadores e Tipos

* **Referência this**
  * **this** é uma referência implícita passada para os métodos, para referenciar o objeto corrente.
  * **this** é usada principalmente em dois contextos:
    * Diferenciar atributos de objetos de parâmetros ou variáveis locais de mesmo nome.
    * Exemplo:

```
public class Caixa {
 private double comprimento, largura, altura;
 public Caixa(double comprimento, double largura, double altura){
     this.comprimento = comprimento;
     this.largura = largura;
     this.altura = altura;
}
```

Identificadores e Tipos

- **Referência this**
  - Acessar o método construtor a partir de outros construtores.
    - Exemplo:

```
public Caixa(){
    this(10, 10, 10)
}
...
}
```

Identificadores e Tipos

* **Referência this**
  * Suponha o seguinte código:

```
Caixa c1 = new Caixa(); // comprimento, largura e altura default = 10
Caixa c2 = new Caixa(10,5,3);
double v1 = c1.volume(); // v1 = 1000
double v2 = c2.volume(); // v2 = 150
```

  * Como o método volume() sabe de qual objeto ele deve obter o tamanho?

Identificadores e Tipos

* **Referência this**
  * **this** é uma referência implícita passada para os métodos, para referenciar o objeto corrente.
    * Usada quando um método precisa se referir ao objeto que o chamou;
    * Pode ser usada dentro de qualquer método para se referir ao objeto corrente;
    * Pode ser usada sempre que uma referência ao objeto for permitida.

Obrigado.

joaopauloaramuni@gmail.com
joaopauloaramuni@fumec.br

UNIVERSIDADE FUMEC
50 Anos