Java e Orientação a Objetos

Instrutor: Eduardo dos Reis Santos

Capitulo 2 – Aula 6 à 10
Estrutura do código Java

Um arquivo de código-fonte (com a extensão .java) contém uma definição de classe. A classe representa uma parte do seu programa e deve ficar dentro de um par de chaves:

```
public class PrimeiroPrograma{
}
```

Tudo o que acontece em Java, acontece dentro de uma classe. Quando você executar um programa em Java, na verdade vai executar uma classe. Então, executar um programa em Java significa informar para a JVM qual classe deve ser carregada para, em seguida, executar o método main desta classe. O main é um método especialmente desenvolvido para "dar partida" na aplicação:

```
public class PrimeiroPrograma{
        public static void main(String[] args) {
        // Seu código entra aqui
        }
}
```

IDENTIFICADORES

Na linguagem de programação Java, um identificador é o nome dado a uma variável, classe ou método.

- O Java case-sensitive, ou seja, faz diferenciação entre letras maiúsculas e minúsculas.
- Não é permitido o uso de palavras reservadas como identificador.

abstract	boolean	break	byte	case	catch
char	class	const	continue	default	do
double	else	extends	final	finally	float
for	goto	if	implements	import	instanceof
int	interface	long	native	new	package
private	protected	public	return	short	static
strictfp	super	switch	synchronized	this	throw
throws	transient	try	void	volatile	while
assert					

Identificadores válidos:

nomeDoUsuario

nome_do_usuario

$variavel

_variavel

Identificadores inválidos:

3idade

nome do usuário

nome-do-usuario

Convenções de código

- Pacotes – Os nomes dos pacotes devem ser substantivos escritos em letras minúsculas.

```
package locadora.cliente
```

- Classes e Interfaces – Os nomes das classes e das interfaces devem ser substantivos, combinando maiúsculas e minúsculas, com a primeira letra de cada palavra em maiúscula. Dentro de cada nome de classe ou interface, as palavras são separadas por letras maiúsculas.

```
class Cliente
interface PessoaFisica
```

- Métodos – Os nomes dos métodos devem ser verbos, combinando maiúsculas e minúsculas, tendo a primeira letra minúscula. Dentro de cada nome de método, as palavras são separadas por letras maiúsculas.

```
cadastrarCliente()
```

- Variáveis – Todas as variáveis devem ter uma combinação de maiúsculas e minúsculas, com a primeira letra em minúscula. As palavras são separadas por letras maiúsculas.

```
clienteAtual
```

- Os nomes das variáveis devem ser significativos e dar uma indicação de seu uso ao leitor. Evite nomes com um único caractere.

Comentários

```
// comentário de uma linha


/* comentário de uma ou mais linhas

*/


/** Comentário para documentação
* que também pode ter uma ou mais linhas
*/
```

Declarando e usando variáveis

Java é fortemente tipado, o que significa que toda variável tem um tipo que não pode ser mudado, uma vez declarado.

```
tipoDaVariavel nomeDaVariavel;


int quantidade;
```

A partir dessa linha de código, a variável quantidade passa a existir na memória e, por ser do tipo **int**, armazena valores inteiros. Para isso, basta atribuir um valor a variável:

```
quantidade = 5;
        ou
int quantidade = 5


// Exibe o conteúdo da variável quantidade
System.out.println( quantidade );
```

Tipos Primitivos

		Valores possíveis				
Tipos	***Primitivo***	***Menor***	***Maior***	***Valor Padrão***	***Tamanho***	***Exemplo***
Inteiro	byte	-128	127	0	8 bits	byte ex1 = (byte)1;
	short	-32768	32767	0	16 bits	short ex2 = (short)1;
	int	-2.147.483.648	2.147.483.647	0	32 bits	int ex3 = 1;
	long	-9.223.372.036.854.770.000	9.223.372.036.854.770.000	0	64 bits	long ex4 = 1l;
Ponto Flutuante	float	-1,4024E-37	3.40282347E + 38	0	32 bits	float ex5 = 5.50f;

	double	-4,94E-307	1.79769313486231570E + 308	0	64 bits	double ex6 = 10.20d;
						ou
						double ex6 = 10.20;

Caractere	char	0	65535	\0	16 bits	char ex7 = 194; ou char ex8 = 'a';
Booleano	boolean	false	true	false	1 bit	boolean ex9 = true;

- Java possui alguns tipos de dados definidos para facilitar a vida do programador. Uma variável de tipo primitivo guarda o valor real informado para ela.

- Por exemplo, int quantidade = 10;, a JVM vai reservar uma determinada quantidade de bytes na memória e o programador vai acessar essa região através do identificador quantidade. O valor dez será copiado para essa região da memória.

- Em Java existem tipos para armazenar valores inteiros e armazenar valores de ponto flutuante.

Tipos inteiros:

byte idade = 30;

short distancia = 31000;

int quantidade = 4552314;

long populacao = 23456787899l;

ou

long populacao = 23456787899L;

Ponto flutuante ou números reais

```
float altura = 1.87F;

float altura = 1.87f;

double peso = 90.35;

boolean maiorIdade = false;

char letra = 'b';
```

Do tipo **boolean** armazena **true** ou **false** e o tipo **char** para armazenar um único caractere entre aspas simples.

O tipo String não é um tipo primitivo, mas sim uma classe que representa uma sequência de caracteres entre aspas duplas:

String nome = “Eduardo”;

Operações Matemáticas em Java

Adição

```
int adicao = 7 + 5;
```

Subtração

```
int subtracao = 7 -5;
```

Multiplicação

```
int multiplicacao = 7 * 5;
```

Divisão

```
double divisao = 5 / 2d; // somente na divisão se coloca o d no final
```

Módulo: Equivale a 5 dividido por 2 que dá 2 com resto 1. O módulo devolve o resto da divisão.

```
int modulo = 5 % 2;
```

Cuidado com o tipo das variáveis para receber o resultado das operações matemáticas.

Casting e Promoção

Quando você declara um tipo para uma variável, temos que tomar que saber se aquele valor cabe para aquela variável:

```
double pi = 3.1415; // compila sem problemas


int numero = pi; // Erro de compilação
```

O erro é exatamente o mesmo, porque pi contém o double (8 bytes) 3.1415 e estamos tentando guardá-lo numa variável do tipo int (4 bytes). Em outras palavras: NÃO CABE!

Outro exemplo:

```
double salario = 500; // Será que compila ?


int numero = salario; // Não compila
```

Isso ocorre porque quando solicitamos que o valor inteiro 500 fosse guardado numa variável do tipo double, ele foi transformado para assumir esse tipo. Essa transformação ocorre automaticamente e é conhecida como promoção.

```
double salario = 500;
int numero = (int) numero; // Casting
```

A tabela abaixo demonstra como utilizar o casting e onde ocorre a promoção.

Para: / De:	byte	short	char	int	long	float	double
byte	----	Promoção	(char)	Promoção	Promoção	Promoção	Promoção
short	(byte)	----	(char)	Promoção	Promoção	Promoção	Promoção
char	(byte)	(short)	----	Promoção	Promoção	Promoção	Promoção
int	(byte)	(short)	(char)	----	Promoção	Promoção	Promoção
long	(byte)	(short)	(char)	(int)	----	Promoção	Promoção
float	(byte)	(short)	(char)	(int)	(long)	----	Promoção
double	(byte)	(short)	(char)	(int)	(long)	(double)	----