Strings e Arrays

Prof. Gustavo Wagner
(Alterações)
Prof. Tiago Massoni
(Slides Originais)

Desenvolvimento de Sistemas

FATEC-PB

Strings

- Seqüências de caracteres
- Não existe tipo primitivo string em Java
- API de Java possui classe chamada String

Gera uma constante e aponta para posição de memória dessa constante

Cria um novo objeto do tipo String e armazena ref. em saudacoes

```
String saudacoes = "Ola";
String saudacoes = new String("Ola");
```

Nova classe Conta

```
public class Conta {
   private String numero;
   private double saldo;
   private Cliente titular;
   ...
}
```

Números podem conter traços, barras

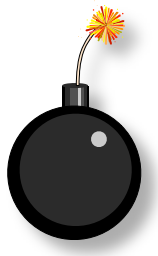
Concatenação de Strings

- Operador '+'
- Qualquer tipo concatenado com uma String é convertido para String

```
int idade = 18;
String minha_idade = idade + " anos";
```

Comparação de Strings

```
String saudacoes = "Ola";
String cumprimento = new String ("Ola");
if (saudacoes == cumprimento)
      //E agora?
```



Comparação de Strings

- Comparar variáveis String não tem resultado desejado (referências)
- Usar método equals() da classe String

```
String saudacoes = "Ola";
String cumprimento = new String ("Ola");
if (saudacoes.equals(cumprimento))
      //Agora sim!!
```

Comparação e tamanho

- boolean equals(umString)
- boolean equalsIgnoreCase(umString)
- int length()

```
String a = "Sharon Stone";
String b = "sharon stone";
int comprimento = a.length();
boolean resposta1 = a.equals(b);
boolean resposta2 = a.equalsIgnoreCase(b);
boolean resposta3 = b.equalsIgnoreCase(a);
```

Qual é o valor de cada resposta?

Classes Wrapper

- Em Java existem classes que encapsulam tipos primitivos (wrappers)
  - Float, Integer, Long
  - Character
  - Double, Float
  - Boolean

Conversões

- Estas classes contém métodos estáticos para transformar uma String num tipo primitivo
  - Integer.parseInt(s)
  - Double.parseDouble(s)
  - Float.parseFloat(s)
- Para o oposto, método estático da classe String
  - String.valueOf(x)

Tratamento de Strings

- String toLowerCase()
- String toUpperCase()
- String trim()

```
String x = " Bom Dia! ";
String y = x.toUpperCase();
String z = x.toLowerCase();
String w = x.trim();
System.out.println(y);
System.out.println(z);
System.out.println(w);
```

```
 BOM DIA!
 bom dia!
Bom Dia!
```

Arrays (Vetores)

- São objetos especiais de Java
- Armazenam dados de um determinado tipo (homogêneo)
- `Tipo[]` é a classe
- Cada componente é identificado por um índice
- O primeiro elemento do array tem índice **0** e o último tem índice

Declaração de arrays

- Se arrays são objetos, o nome do array é uma variável referência para ele

  `Tipo[] nomeArray;`

```
int [] meuArrayInteiros;

char [] stringAntigaemC;

Double arrayComOutroNome[];
```

Criação de arrays

- O Operador **`new TipoArray[tamanho]`** cria um objeto array, com valores padrão (zerados)
- O comprimento do array é acessível pelo atributo final e público **`length`**
- Arrays têm tamanho fixo depois

Declaração, criação,

```
int[] a;
double[] x = {10.0, 15.0, 20.0};


a = new int[100];

double[] salarios = new double[5];

for (int i = 0; i < salarios.length; i++) {
    salarios[i] = i * 1000;
}
```

Forma mais direta de inicialização

Inicialização com laço

Acesso a elementos do array

variável[expressão_inteira]

- Escrita e leitura a elementos do array é feito através de índices

```
double [] salarios = {20.0,30.0,50.0};
salarios[0] = salarios[0] * 1000;
System.out.println(salarios[2]);
```

Acesso inválido

- Se é feito acesso a um elemento indefinido de um array, é gerada uma exceção:
  - IndexOutOfBoundsException

```
int [] numeros = {2,4};
System.out.println(numeros[5]);
```

Gera um erro em tempo de execução

Arrays de Objetos

- Armazenam referências a objetos de um determinado tipo
- Devem ser alocados da mesmo forma que arrays de tipos primitivos
- Valor default: cada elemento do array é inicializado com null

Arrays de Objetos

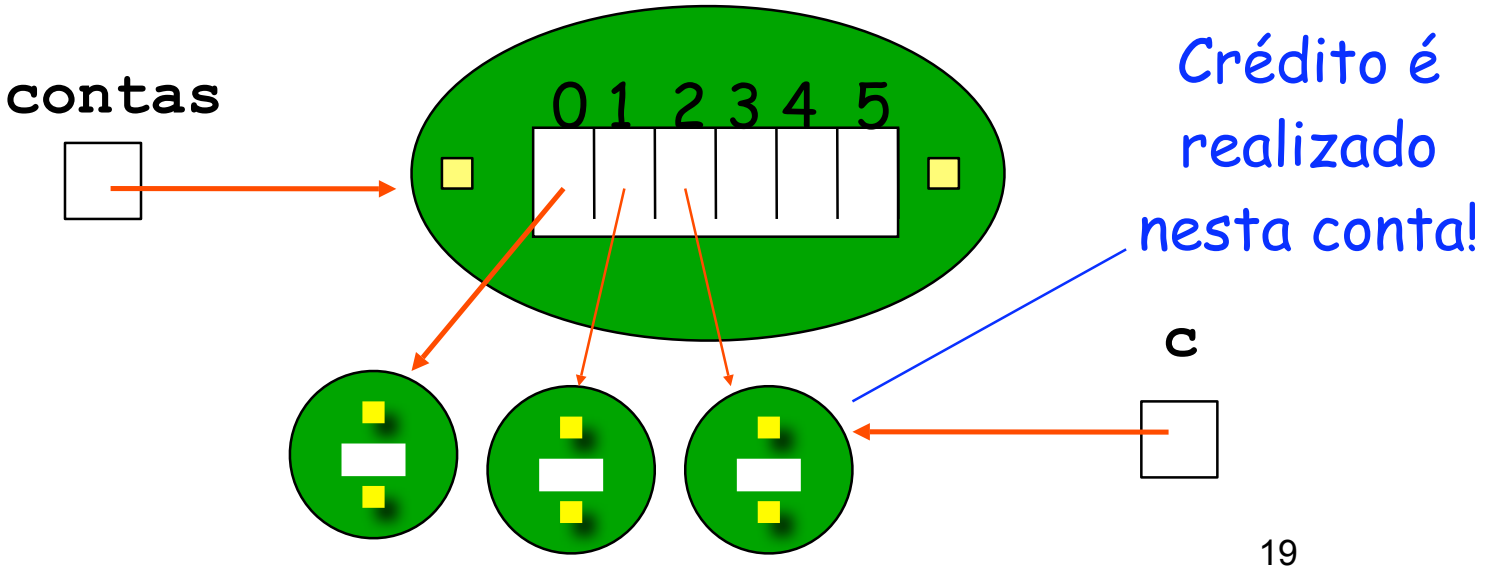
```
Conta[] contas = new Conta[10];
...
contas[1] = new Conta (123-4,20.0);
if (contas[1].retornarSaldo() == 20.0)
     //faz alguma coisa
```

Representa o envio da mensagem retornarSaldo() para um objeto referenciado pela 2a. posição do array contas

Atualização de objetos em

- *Aliasing* garante modificação dos objetos do array

```
Conta c = contas[2];
c.creditar(20);
```



Aula Prática
Strings e Arrays

Atividades

- Importar resultados da aula anterior para um novo projeto
- Criar classe BibliotecaMidias
- Um objeto desta classe servirá como um banco de mídias (Livros, Filmes, Músicas, etc) que oferece serviços da biblioteca de mídias (ver, pesquisar, abrir, etc)

Atividades

- Esta classe vai ter um atributo para cada tipo de mídia: o array que armazenará as músicas, outro que armazenará os filmes, etc, e índices mostrando a próxima posição livre para cada array;
- Implementar os serviços de uma coleção
  - Inserir música, inserir livro, etc;
  - criar métodos que retornam a quantidade de cada tipo de mídias;

Classe BibliotecaMidias

```
public class BibliotecaMidias {
  BibliotecaMidias(){}
  void inserirMusica(Musica musica){}
  void removerMusica(String codigoMusica){}
  Musica procurarMusica (String nomeMusica)
{}
Musica procurarMusicaPorCodigo (String
codigoMusica) {}
  int quantidadeMusicas(){}
  void inserirLivro(Livro livro){}
  void removerLivro(String codigoLivro){}
...
}
```

Todos os métodos são public!

Atividades

- Dica: implementar primeiro o método procurar (ele será utilizado por outros métodos)
  - Retorna a mídia que tem o código passado como parâmetro, ou retorna null se o número não for encontrado
- Método remover() não usa procurar() – pense porquê!

Atividades

- Ao terminar, declarar uma classe programa que cria um BibliotecaMidias e oferece serviços ao usuário
  - Se usuário quer cadastrar uma mídia (Livro, Música, etc), pedir os argumentos necessários.
  - Etc...