

Tipos referência, Strings e Arrays

Tipos referência

- Em Java há dois conjuntos de tipos
 - Tipos primitivos
 - Tipos referência
- Tipos primitivos
 - int, long, double, float, ...
- Tipos referência
 - classes, interfaces e arrays

Tipos de referência

- Em Java não se trabalha diretamente com os objetos e sim com

referências a objetos

- Isso tem implicações na maneira em que objetos são comparados e copiados

Strings

Strings

- São seqüências de caracteres
- Não há um tipo primitivo para Strings em Java
- Em Java, Strings são **objetos**

```
String mensagem = "Operação concluída com sucesso";
```

Aqui Java cria um novo objeto do tipo String e o armazena na variável mensagem

Concatenação de Strings

- Operador **+** é usado para concatenação

```
String nome = "George";  
String sobrenome = "Bush";  
String nomeCompleto = nome + " " + sobrenome;
```

```
int anos = 10;  
double rendimento = 1270.49;
```

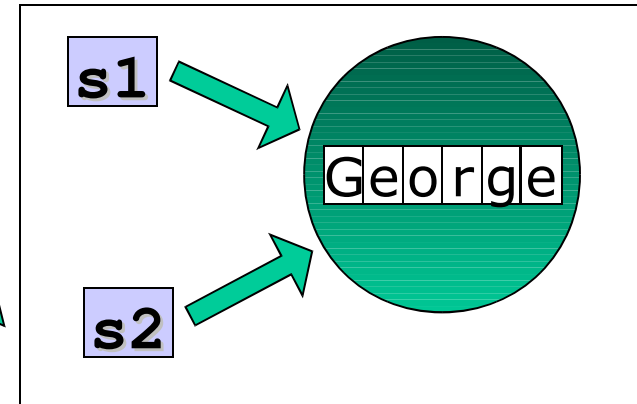
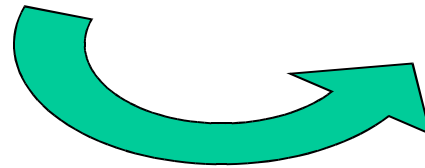
```
String s = "Em " + anos + " anos o " +  
    "rendimento será de " + rendimento;
```

```
System.out.println(s);
```

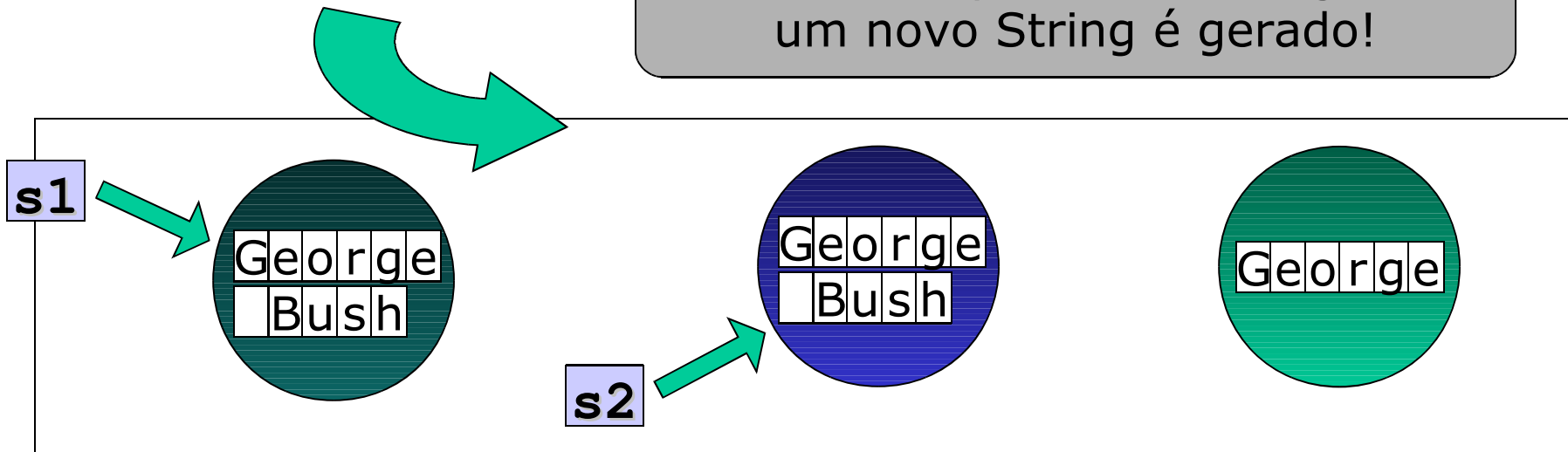
A conversão para
String é feita
automaticamente

Strings são tipos referência

```
String s1 =  
    "George";  
String s2 =  
    "George";  
s1 = s1 + " Bush";  
s2 = s2 + " Bush";
```



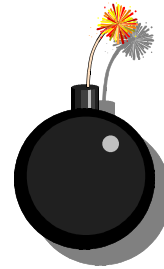
Ao manipular um String,
um novo String é gerado!



Igualdade de Strings

- Para testar se dois Strings são iguais, não deve ser usado `==`
- Deve-se usar o método `equals`:

`s1 == s2`



`s1.equals(s2)`



Igualdade de Strings

```
//Cria dois novos Strings
String s1 = "George";
String s2 = "George";

// Nesse momento, s1==s2 é verdadeiro!

s1 = s1 + " Bush"; // Cria um novo string e o atribui para
s1
s2 = s2 + " Bush"; // Cria um novo string e o atribui para
s2

if (s1 == s2)
    System.out.println("s1 e s2 sao os mesmos objetos.");
else
    System.out.println("s1 e s2 NAO sao os mesmos objetos.");

if (s1.equals(s2))
    System.out.println("s1 e s2 possuem o mesmo conteúdo.");
else
    System.out.println("s1 e s2 NAO possuem o mesmo
                        conteúdo.");
```

Strings: Comparação e comprimento

- **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 o valor das respostas?

String: tratamento

- **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!
```

Strings: índices e substrings

Índices em Java
começam a partir de 0

Retorna um substring de
indiceInício até **indiceFinal-1**

- `int indexOf(umString)`
- `String substring(int indiceInício, int indiceFinal)`
- `char charAt(int indice)`

```
String x = "Pernambuco";  
String y = x.substring(0,5);  
String z = x.substring(6,10);  
int indice = x.indexOf("na");  
char letra = x.charAt(5);  
System.out.println(x); System.out.println(y);  
System.out.println(z); System.out.println(indice);  
System.out.println(letra);
```

Qual é a saída?

Arrays

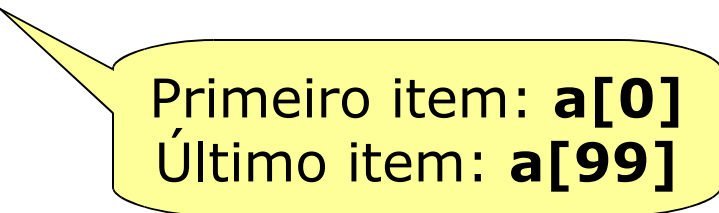
Arrays

- São tipos especiais de Java. Objetos especiais
- Arrays também são tipos referência
- Todos os elementos de um array são do mesmo tipo
- Arrays têm tamanho fixo depois de criados

Declaração e criação de arrays

```
int[] a;  
double[] x;  
Cliente[] clientes;
```

```
int[] a = new int[100];
```



Primeiro item: **a[0]**
Último item: **a[99]**

```
String[] nomes = new String[200];
```

Inicialização de Arrays

```
int[] primosPequenos = {2, 3, 5, 7, 11,  
    13};
```

```
String[] cores = {"Vermelho", "Azul", "Amarelo"};
```

```
double[] salarios = new double[5];
```

```
for (int i = 0; i<5; i++) {  
    salarios[i] = i * 1000;  
}
```


Arrays multidimensionais

```
int[][] matriz;
```

Declaração não especifica dimensões

```
int[][] matriz = new int[10][5];
```

```
for (int i=0; i<10; i++)  
    for (int j=0; j<5; j++)  
        matriz[i][j] = 100;
```

Cria e inicializa um array bidimensional

```
long[][] x = { {0,1}, {2,3}, {4,5} };
```

x[0][0]

x[0][1]

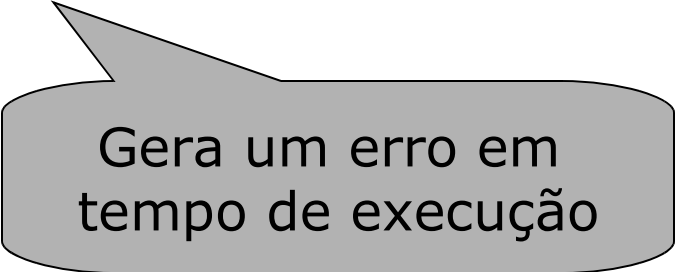
x[2][0]

Cria um array de 3 por 2

Acesso inválido

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

```
String nomes[] = {"José", "João", "Maria"};  
System.out.println(nomes[5]);
```



Gera um erro em tempo de execução

Coleção de contas com array

- Uma classe que guarda contas num array de contas
 - Repositorio ou conjunto de contas
- Métodos para inserção, procura, remoção e atualização dos objetos

RepositorioContasArray

```
public class RepositorioContasArray {  
    private Conta[]  contas;  
    private int indice;  
    private final static int tamCache = 100;  
  
    public RepositorioContasArray() {  
        indice = 0;  
        contas = new Conta[tamCache];  
    }  
  
    public void inserir(Conta c){  
        contas[indice] = c;  
        indice = indice + 1;  
    }  
  
    ...  
}
```

RepositorioContasArray

```
private int procurarIndice(String num) {  
    int      i = 0;  
    int      ind = -1;  
    boolean achou = false;  
  
    while ((i < indice) && !achou) {  
        if ((contas[i].getNumero()).equals(num)) {  
            ind = i;  
            achou = true;  
        }  
        i = i + 1;  
    }  
    return ind;  
}
```

RepositorioContasArray

```
public boolean existe(String num) {  
    boolean resp = false;  
  
    int i = this.procurarIndice(num);  
    if(i != -1){  
        resp = true;  
    }  
  
    return resp;  
}
```

RepositorioContasArray

```
public void atualizar(Conta c){
    int i = procurarIndice(c.getNumero());
    if (i != -1) {
        contas[i] = c;
    } else {
        System.out.println("Conta nao encontrada");
    }
}
```

```
public Conta procurar(String num){
    Conta c = null;
    if (existe(num)) {
        int i = this.procurarIndice(num);
        c = contas[i];
    } else {
        System.out.println("Conta nao encontrada");
    }
    return c;
}
```

RepositorioContasArray

```
public void remover(String num){
    if (existe(num)) {
        int i = this.procurarIndice(num);
        contas[i] = contas[indice - 1];
        contas[indice - 1] = null;
        indice = indice - 1;
    } else {
        System.out.println("Conta nao encontrada");
    }
}
```