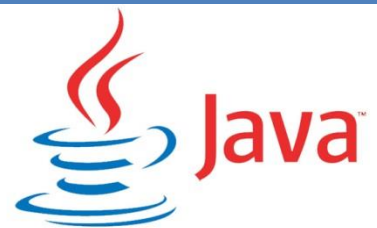


Lógica de Programação

Unidade 8 – Método toString



QI ESCOLAS E FACULDADES
Curso Técnico em Informática

SUMÁRIO

INTRODUÇÃO	3
MÉTODO TOSTRING	3
SINTAXE DO MÉTODO TOSTRING	3
<i>Exemplo de implementação do toString</i>	<i>3</i>
EXEMPLO UML DE CLASSE COM TOSTRING.....	3
EXEMPLO DE SINTAXE DA CLASSE COM TOSTRING	4

INTRODUÇÃO

O objeto, além de executar tarefas (métodos), também armazena dados (seus atributos). Estes atributos, em determinados momentos, precisam ser exibidos para o usuário. Para facilitar esta exibição, o Java possui um método especial, denominado **toString**, que pode ser implementado na classe para criar um padrão de exibição do estado do objeto.

MÉTODO TOSTRING

Implementamos o método **toString** para retornar o objeto em formato de **texto**. Ele simplifica a exibição dos atributos do objeto, convertendo o objeto para texto. Neste método, determinamos como os atributos devem ser exibidos.

Sintaxe do método toString

Este método não pode ser criado de qualquer maneira. Ele possui uma sintaxe padrão, onde alteramos apenas o que vai no “return”. O nome deve ser **toString**, sempre deve retornar uma **String** e não possui argumentos.

```
public String toString() {
    return "mensagem";
}
```

Exemplo de implementação do toString

```
public String toString() {
    return "Nome: " +this.nome+ "\nIdade: " +this.idade+ "\nPeso: " +this.peso;
}
```

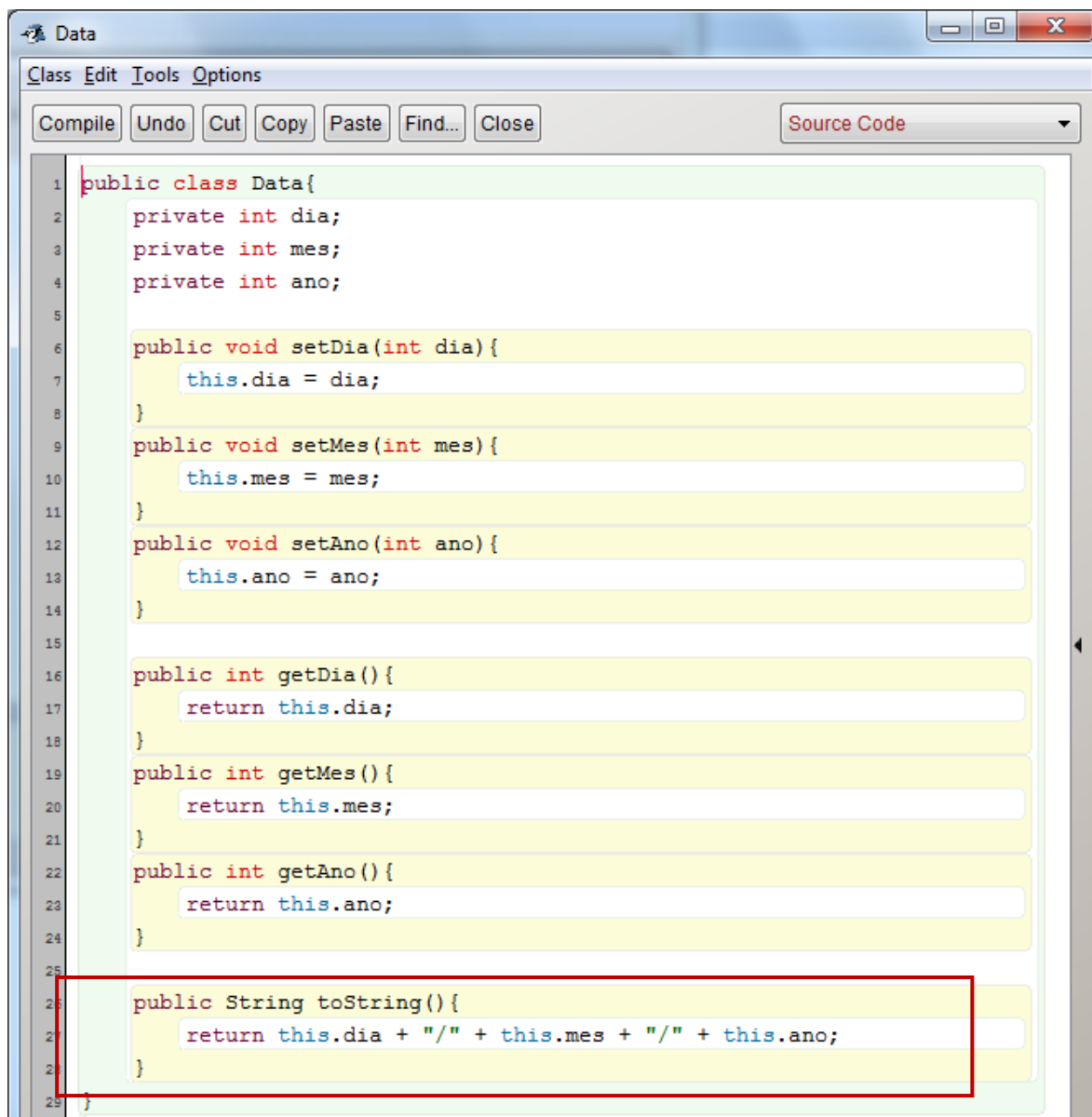
No exemplo, no *return* especificamos como o objeto será apresentado. Aqui acontece o processo de **concatenação**, ou junção de texto. Quando trabalhamos com textos, o símbolo + significa concatenação (junção). O que fica entre aspas é o texto que acompanha o atributo. O \n significa “nova linha” (opcional).

Exemplo UML de classe com toString

Data
-dia: byte -mes: byte -ano: int
+getDia():byte +getMes():byte +getAno():int +setDia(dia:byte):void

```
+setMes(mes:byte):void
+setAno(ano:byte):void
+toString():String
```

Exemplo de Sintaxe da Classe com toString



Observe no destaque a implementação do método **toString**. Como estaremos exibindo uma data, foi concatenada uma barra entre os atributos, para que a data aparecesse no formato "xx/xx/xxxx".

Vejamos como fica a classe Main com o método `toString` na classe.

```

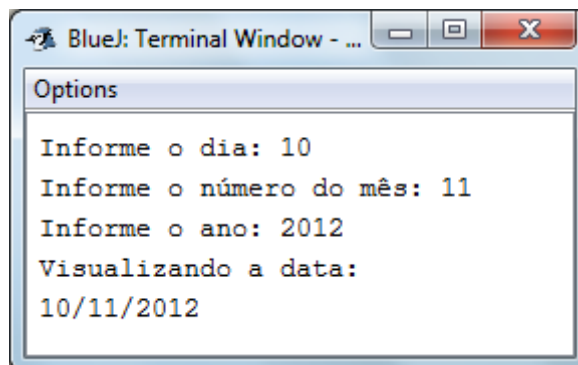
1 import java.util.Scanner;
2 public class Main{
3     public static void main(String args[]){
4         Scanner ler = new Scanner(System.in);
5         Data d1 = new Data();
6
7         //O usuário informa a data
8         System.out.print("Informe o dia: ");
9         d1.setDia(ler.nextInt());
10        System.out.print("Informe o número do mês: ");
11        d1.setMes(ler.nextInt());
12        System.out.print("Informe o ano: ");
13        d1.setAno(ler.nextInt());
14
15        System.out.println("Visualizando a data:");
16        System.out.print(d1);
17    }
18 }

```

Observe que no código acima, linha 16, colocamos apenas o objeto no comando *System.out.print*. Observe que **não é preciso invocar o método toString**. Mas, se quisermos, também funciona, porém é desnecessário:

```
System.out.print(d1.toString());
```

Observe o resultado no terminal:



```

Blue: Terminal Window - ...
Options
Informe o dia: 10
Informe o número do mês: 11
Informe o ano: 2012
Visualizando a data:
10/11/2012

```