Programação Orientada a Objetos – Java – classe *Calendar e Date*

Prof. Edwar Saliba Júnior – versão 1.0

Exemplo de Utilização das Classes *Calendar* e *Date*

Há algum tempo, na programação em Java, utilizava-se para manipular datas, única e exclusivamente, a classe *Date*. Porém, com a evolução da linguagem esta classe ficou obsoleta e foi então criada a classe *Calendar*.

Vamos ver um exemplo de como trabalhar com a classe *Calendar* e também que apesar de estar obsoleta, a classe *Date* ainda funciona. Porém, seu uso não deve ser incentivado.

```
1  package dataemjava;
2
3  import java.util.Calendar;
4  import java.util.Date;
5
6  /**
7   * Exemplo de utilização da classe Calendar.
8   * @author Edwar Saliba Júnior
9   */
10 public class Main {
11
12     public static void main(String[] args) {
13         Date d, conv;
14         Calendar c;
15
16         System.out.print("\n\n======= Date =======\n\n");
17         d = new Date(111, 9, 3);
18         System.out.print(d);
19
20         System.out.print("\n\n======= Date (Data e Hora) =======\n\n");
21         d = new Date(111, 9, 3, 15, 12, 53);
22         System.out.print(d);
23
24         System.out.print("\n\n======= Calendar =======\n\n");
25         c = Calendar.getInstance();
26         c.set(2011, 9, 3);
27         System.out.println(c.getTime());
28
29         System.out.print("\n\n======= Calendar (Data e Hora) =======\n\n");
30         c.set(2011, 9, 3, 15, 12, 53);
31         System.out.println(c.getTime());
32
33         System.out.print("\n\n======= Conversão Calendar para Date =======\n\n");
34         conv = c.getTime();
35         System.out.println(conv);
36
37         System.out.print("\n\n");
38     }
39 }
```

Figura 1: Exemplo de Utilização de Date e Calendar

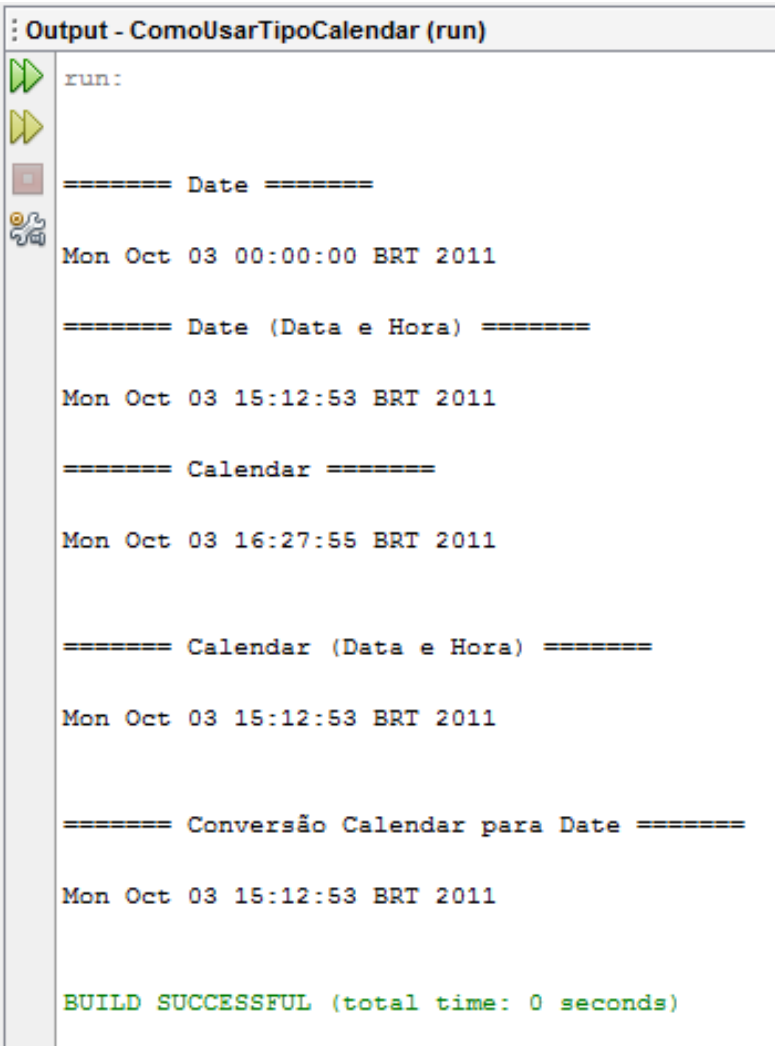

Figura 2: Resultado da Execução do Programa da Figura 1

A classe Date

Na Figura 1 podemos observar um programa que utiliza tanto o tipo *Date* quanto o tipo *Calendar*. Se observarmos as linhas 17 e 21 veremos que na chamada do método construtor a palavra *Date* foi tachada pela IDE NetBeans, isto devido ao fato do comando estar obsoleto e seu uso ser desaconselhável.

Duas pequenas observações sobre a classe *Date*:

- o construtor recebe três parâmetros: ano, mês e dia. Sendo que deve-se diminuir 1900 do ano passado como parâmetro, ou seja, neste caso como queríamos imprimir o ano de 2011, então foi passado como parâmetro o número 111, pois, 2011 - 1900 = 111;

- os números que representam os meses do ano vão de 0 (zero) até 11 (onze), ou seja, o mês de Janeiro é representado pelo número zero e o mês de Dezembro pelo número onze.

A classe *Date* possui mais de um construtor. O primeiro nós já vimos na linha 17 da Figura 1, este é um construtor que recebe como parâmetro o ano, o mês e o dia. Um segundo construtor pode ser visto na linha 21, onde são passados como parâmetros: ano, mês, dia, hora, minutos e segundos. No caso de usarmos o primeiro construtor, então as horas, minutos e segundos serão inicializadas com zero. Estas saídas podem ser observadas na Figura 2.

Trabalhando com a classe Calendar

A classe *Calendar* veio substituir a já ultrapassada classe *Date*.

No código apresentado na Figura 1 podemos ver que foi criada uma variável chamada de “c” do tipo *Calendar* (linha 14).

O tipo *Calendar* nos permite fazer diversas operações com datas. Contudo, neste tutorial será mostrado apenas o básico.

Ao criarmos uma variável do tipo *Calendar*, devemos instanciá-la utilizando o método *getInstance* da própria classe *Calendar*, como pode ser visto na linha 25 da Figura 1. Ao fazermos tal instanciação, estamos ao mesmo tempo, criando o objeto e o inicializando com os valores (data e hora) do sistema operacional.

Para atribuirmos um novo valor de data ou de data e hora para um objeto do tipo *Calendar*, basta utilizarmos o método *set* da classe. Conforme mostrado na linha 26 da Figura 1. Nesta linha estamos atribuindo a data de 03 de Outubro de 2011 ao objeto “c”. E na linha 27 estamos imprimindo a data que acabamos de atribuir ao objeto, juntamente com a hora, minutos e segundos capturados no momento de sua criação.

Repare que ao atribuirmos a data para o objeto “c” na linha 26 da Figura 1, estamos passando os seguintes valores para o método *set*: ano (2011), mês (09, que neste caso se refere ao mês de Outubro, visto que os meses têm sua representação numérica iniciada em zero e terminada em onze) e dia (03).

Para imprimirmos a data de um objeto *Calendar*, basta chamarmos o método *getTime* da classe. Este método retornará um objeto do tipo *Date* que será interpretado pela classe *System* e impresso em forma de *String* (conforme pode ser observado na Figura 2).

Na linha 30 da Figura 1 temos uma atribuição de data (ano, mês e dia) e hora (horas, minutos e segundos), nesta ordem, para o objeto “c”. E na linha 34 temos um exemplo de conversão do tipo *Calendar* para o tipo *Date*.

Algumas Operações com Calendar

Na Figura 3 podemos observar diversas operações com a classe *Calendar* (todas comentadas) e na Figura 4 podemos observar os resultados da execução do *software* apresentado na Figura 3.

```java
package dataemjava;

import java.util.Calendar;

/**
 * Exemplo de operações com Calendar.
 * @author Edwar Saliba Júnior
 */
public class Main {

    public static void main(String[] args) {
        Calendar c, f;
        int dia, mes, ano;

        System.out.print("\n\n======= Instanciando e Inicializando os Objetos =======\n\n");
        c = Calendar.getInstance();
        f = Calendar.getInstance();

        c.set(2011, 9, 3, 0, 0, 0);    //  03/10/2011
        f.set(2011, 9, 2, 0, 0, 0);    //  02/10/2011

        System.out.println("c = " + c.getTime());
        System.out.println("f = " + f.getTime());

        // Desmembrando a data.
        System.out.print("\n\n======= Desmembrando a Data =======\n\n");

        dia = f.get(Calendar.DATE);
        mes = f.get(Calendar.MONTH);
        ano = f.get(Calendar.YEAR);

        System.out.println("Dia: " + dia + "  Mês: " + mes + "  Ano: " + ano);

        // Imprimindo a constante que representa o número do mês de Novembro.
        System.out.print("\n\n======= Constante que representa o mês de Novembro =======\n\n");

        System.out.println("Novembro: " + Calendar.NOVEMBER);

        // Acrescentando dias, meses e anos a uma data.
        System.out.print("\n\n======= Adicionar dias, meses e anos =======\n\n");
        c.add(Calendar.DATE, 7); // Aumentando 7 dias.
        System.out.println(c.getTime());
        c.add(Calendar.MONTH, 2); // Aumentando 2 meses.
        System.out.println(c.getTime());
        c.add(Calendar.YEAR, 1); // Aumentando 1 ano.
        System.out.println(c.getTime());

        // Acrescentando horas, minutos e segundos.
        System.out.print("\n\n======= Adicionar horas, minutos e segundos =======\n\n");
        c.add(Calendar.HOUR, 3); // Aumentando 3 horas.
        System.out.println(c.getTime());
        c.add(Calendar.MINUTE, -15); // Diminuindo 15 minutos.
        System.out.println(c.getTime());
        c.add(Calendar.SECOND, 12); // Aumentando 12 segundos.
        System.out.println(c.getTime());

        // Comparando datas.
        System.out.print("\n\n======= Comparando Datas =======\n\n");
        c = Calendar.getInstance();
        f = Calendar.getInstance();
        System.out.println("c = " + c.getTime());
        System.out.println("f = " + f.getTime());
        System.out.println("Resultado comparação quando \"c\" == \"f\": " + c.compareTo(f));
        f.add(Calendar.DATE, 2);
        System.out.println("c = " + c.getTime());
        System.out.println("f = " + f.getTime());
        System.out.println("Resultado comparação quando \"c\" < \"f\": " + c.compareTo(f));
        f.add(Calendar.DATE, -4);
        System.out.println("c = " + c.getTime());
        System.out.println("f = " + f.getTime());
        System.out.println("Resultado comparação quando \"c\" > \"f\": " + c.compareTo(f));

        System.out.print("\n\n");
    }
}
```

Figura 3: Algumas Operações com a Classe Calendar

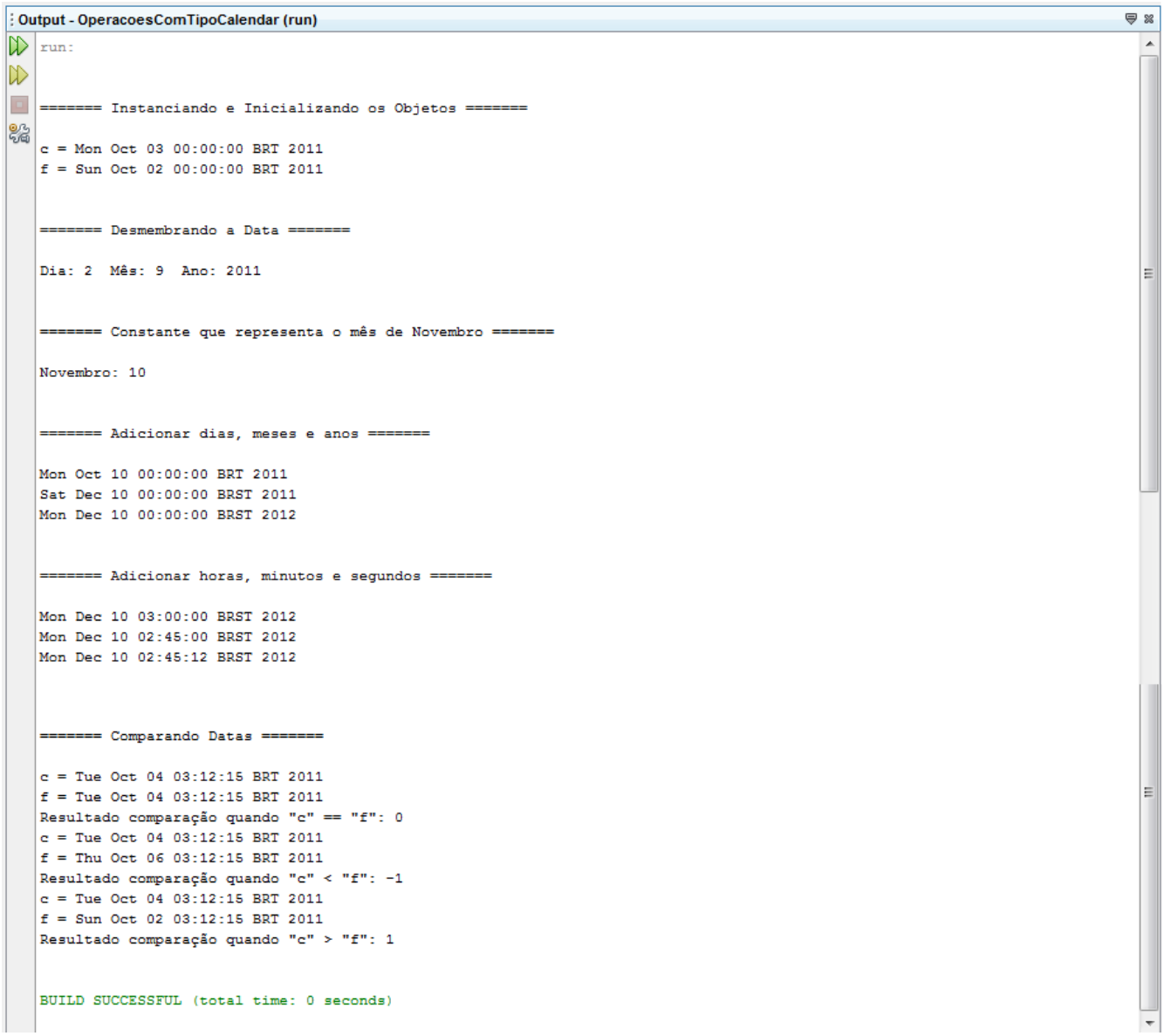

Figura 4: Resultado da Execução do Software da Figura 3