Artigo

Trabalhando com as classes Date, Calendar e SimpleDateFormat em Java

Veja neste artigo como manipular as classes Date, Calendar e SimpleDateFormat, a converter Strings para tipos data e vice versa. Aprenda a internacionalizar as datas de acordo com a região e saiba algumas particularidades dessas classes.

As classes apresentadas neste artigo demonstram como a programação em Java consegue manipular as datas, horas e internacionalizá-las. No Java o tempo é representado em milissegundos, sendo medido a partir da data 01/01/1970. Essa data pode ser recuperada através do comando `System.currentTimeMillis()`.

Abaixo são mostradas as principais classes utilizadas quando se trata de algum dado que envolva datas.

Classe Date

A data representa o tempo, um tempo é composto por ano, mês, dia atual, minuto atual, entre outras propriedades que essa classe possui.

Hoje a maioria dos métodos da **classe Date** estão classificados como deprecated (depreciados), ou seja, são métodos que não são mais utilizados, por isso essa classe foi substituída pela classe `Calendar`, para haver suporte correto à internacionalização do sistema de datas.

Saiba mais: **Classe Calendar**

Na **Listagem 1** é instanciada a classe Date e exibida a data atual em detalhes.

```
1 import java.util.Date;
2
3 public class Testa_Date {
4     public static void main(String[] args) {
5         Date data = new Date();
6         System.out.println("Data Agora: "+data);
7     }
8 }
```

Listagem 1. Testando a classe Date

Classe Calendar

Essa classe pode produzir os valores de todos os campos de calendário necessários para implementar a formatação de data e hora, para uma

A classe Calendar é a mais usada quando se trata de datas, mas como é uma classe abstrata, não pode ser instanciada, portanto para obter um calendário é necessário usar o método estático `getInstance()`, apresentado no exemplo da **Listagem 2.**

```
1 import java.util.Calendar;
2
3 public class Data_Calendar{
4
5     public static void main(String[] args) {
6         Calendar c = Calendar.getInstance();
7         System.out.println("Data e Hora atual: ”+c.getTime());
8     }
9 }
```

Listagem 2. Recuperação da data com a classe Calendar

Com a classe Calendar também se consegue manipular a data e hora com os métodos que são fornecidos, abaixo seguem os exemplos.

```
1  import java.util.Calendar;
2
3  public class Testa_Metodo_Get_Calendar{
4
5      public static void main(String[] args) {
6          Calendar c = Calendar.getInstance();
7
8          System.out.println("Data/Hora atual: "+c.getTime());
9          System.out.println("Ano: "+c.get(Calendar.YEAR));
10         System.out.println("Mês: "+c.get(Calendar.MONTH));
11         System.out.println("Dia do Mês: "+c.get(Calendar.DAY_OF_MONTH));
12     }
13 }
```

```
import java.util.Calendar;

public class Testa_Metodos_Set_Get_Calendar{

    public static void main(String[] args) {
        Calendar c = Calendar.getInstance();
        c.set(Calendar.YEAR, 1995);
        c.set(Calendar.MONTH, Calendar.MARCH);
        c.set(Calendar.DAY_OF_MONTH, 20);

        System.out.println("Data/Hora atual: "+c.getTime());
        System.out.println("Ano: "+c.get(Calendar.YEAR));
        System.out.println("Mês: "+c.get(Calendar.MONTH));
        System.out.println("Dia do Mês: "+c.get(Calendar.DAY_OF_MONTH));
    }

}
```

Listagem 4. Alterando a data/hora com método set

Na **Listagem 5**, mostra-se como recuperar a hora do dia, fazendo uma interação com o usuário informando uma mensagem de boas vindas.

```
import java.util.Calendar;

public class Msg_Boas_Vindas_Calendar{

    public static void main(String[] args) {
        Calendar c1 = Calendar.getInstance();
        int hora = c1.get(Calendar.HOUR_OF_DAY);

if(hora > 6 && hora < 12){
            System.out.println("Bom Dia");
        }else if(hora > 12 && hora < 18){
            System.out.println("Boa Tarde");
        }else{
            System.out.println("Boa Noite");
        }
    }
}
```

Para mais informações sobre a classe Calendar, acesse o link para as **Documentações Oracle.**

DateFormat

Essa classe permite converter informações do tipo String para data do tipo Date, permitindo seguir um formato. Consegue-se trabalhar ao contrário, convertendo um dado do tipo Date para uma String. Por ser uma classe abstrata, não é possível instanciá-la, por isso deve ser usado para método estático `getDateInstance()`. Sempre quando declarado é preciso importar o pacote `java.text`.

Abaixo são mostrados alguns exemplos dos métodos de formatação das datas.

```
1  import java.util.Calendar;
2  import java.text.DateFormat;
3  import java.util.Date;
4
5  public class Formatando_Datas{
6
7      public static void main(String[] args) {
8          Calendar c = Calendar.getInstance();
9          c.set(2013, Calendar.FEBRUARY, 28);
10         Date data = c.getTime();
11         System.out.println("Data atual sem formatação: "+data);
12
13         //Formata a data
14         DateFormat formataData = DateFormat.getDateInstance();
15         System.out.println("Data atual com formatação: "+ formataData.format(da
16
17         //Formata Hora
18 DateFormat hora = DateFormat.getTimeInstance();
19         System.out.println("Hora formatada: "+hora.format(data));
20
21         //Formata Data e Hora
22         DateFormat dtHora = DateFormat.getDateTimeInstance();
23         System.out.println(dtHora.format(data));
24
25     }
```

Listagem 6. Formatando data atual

```
import java.util.Calendar;
import java.text.DateFormat;
import java.util.Date;

public class Formatando_Saida_Datas{

    public static void main(String[] args) {
        Calendar c = Calendar.getInstance();
        Date data = c.getTime();

        DateFormat f = DateFormat.getDateInstance(DateFormat.FULL); //Data COmp
        System.out.println("Data brasileira: "+f.format(data));

f = DateFormat.getDateInstance(DateFormat.LONG);
        System.out.println("Data sem o dia descrito: "+f.format(data));

        f = DateFormat.getDateInstance(DateFormat.MEDIUM);
        System.out.println("Data resumida 1: "+f.format(data));

        f = DateFormat.getDateInstance(DateFormat.SHORT);
        System.out.println("Data resumida 2: "+f.format(data));
    }
}
```

Listagem 7. Formatações de datas

Conversões

Às vezes é preciso transformar um texto em uma data ou vice versa, para isso abaixo é exibida a função `parse` e a classe `SimpleDateFormat`. Geralmente a classe `SimpleDateFormat` é mais usada quando trata-se de formatação de datas, pois já no seu construtor, quando instanciada, permite passar como argumento o formato da data desejada, como apresentada nos exemplos abaixo.

```java
import java.util.Calendar;
import java.text.DateFormat;
import java.util.Date;
import java.text.SimpleDateFormat;
import java.text.ParseException;

public class Conversao_Datas{

    public static void main(String[] args) throws ParseException{
        Calendar c = Calendar.getInstance();
        Date data = c.getTime();
        DateFormat f = DateFormat.getDateInstance();

        Date data2 = f.parse("12/01/1995");
        System.out.println(data2);

        SimpleDateFormat sdf = new SimpleDateFormat("dd/MM/yyyy");
        System.out.println("Data formatada: "+sdf.format(data));

        //Converte Objetos
        System.out.println("Data convertida: "+sdf.parse("02/08/1970"));
    }
}
```

Listagem 8. Convertendo Datas com método parse e a classe `SimpleDateFormat`

Para mais detalhes, acesse o link para as **Documentações Oracle.**

Saiba mais: **Boas práticas ao trabalhar com datas**

Internacionalização das datas

O Java oferece a classe `Locale`, que permite definir de qual país deseja-se obter o retorno das informações, como data e hora. Nos exemplos acima não foi

computador quais são as configurações regionais. Na **Listagem 9** é mostrado como formatar data e hora de acordo com outros países.

```
1  import java.util.Calendar;
2  import java.util.Locale;
3  import java.text.DateFormat;
4  import java.text.ParseException;
5  import java.util.Date;
6
7  public class Locale_Datas{
8
9      public static void main(String[] args) throws ParseException {
10         Calendar c = Calendar.getInstance();
11         Date data = c.getTime();
12
13         Locale brasil = new Locale("pt", "BR"); //Retorna do país e a língua
14         Locale eua = Locale.US;
15         Locale italia = Locale.ITALIAN;
16
17         DateFormat f2 = DateFormat.getDateInstance(DateFormat.FULL, brasil);
18         System.out.println(“Data e hora brasileira: ”+f2.format(data));
19
20         DateFormat f3 = DateFormat.getDateInstance(DateFormat.FULL, eua);
21         System.out.println(“Data e hora americana: ”+f3.format(data));
22
23         DateFormat f4 = DateFormat.getDateInstance(DateFormat.FULL, italia);
24         System.out.println(“Data e hora italiana: ”+f4.format(data));
25     }
26 }
```

Listagem 9. Internacionalizando datas

Veja neste link a lista dos códigos referentes aos outros países, que podem ser usados como argumentos nos construtores da classe `Locale` . **ISO-3166 Country Codes**