

# Classe Date

A classe Date é uma das mais simples do Java, com ela é possível trabalhar com funções referentes ao tempo, sendo eles:

- Segundos;
- Minutos;
- Horas;
- Dias;
- Semanas;
- Meses;
- Anos...

Essa classe guarda a informação em milissegundos, o tipo é um **long**. Para quem não sabe long é um tipo de variável que armazena apenas números inteiros. Diferente do **int** essa variável tem a capacidade de armazenar valores inteiros maiores que um int.

As datas como são exibidas em milissegundos deve ter como base uma data inicial, essa data é 01/01/1970. Sendo assim se você for obter a hora atual serão contabilizados os milissegundos a partir do dia 01/01/1970.

Vamos para um pequeno exemplo da classe Date:

```
//Importando classe
import java.util.Date;

public class Teste {

    public static void main(String[] args) {

        //Instanciar objeto
        Date d = new Date();

        //Exibir dia
        System.out.println(d.getDate());

    }

}
```

Note que na imagem acima foi instanciado um objeto da classe **Date** e quando chamado o método **getDate()** ele está com um risco.

Sempre que você notar esse risco o Java considera esse comando *deprecated* ou em sua tradução comando em desuso.

A classe **Date** é muito comentada, porém à partir do Java 8 essa classe deixa de ser utilizada devido à demora no processamento e a dificuldade em trabalhar com os fuso-horários.

Então se por ventura você estiver utilizando a versão 8 ou superior do Java não utilize a classe **Date**, vamos ver o que veio de novo para suprir essa classe ;)

# Classe Local Date-Time

Vamos ao exemplo mais simples da classe que trabalha com tempo. Lembre-se de que há muitas funções e no final desse arquivo você poderá estudar cada uma delas nos sites indicados, porém sempre pesquise pelas novidades que o Java traz.

Estrutura básica retornando data e hora completas:

```
//Importando classe
import java.time.LocalDateTime;

public class Teste {

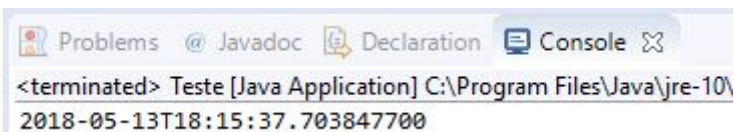
    public static void main(String[] args) {

        //Instanciar objeto
        LocalDateTime tempo = LocalDateTime.now();

        //Exibindo a data e a hora atual
        System.out.println(tempo);

    }

}
```



The screenshot shows the IDE's console window with tabs for Problems, Javadoc, Declaration, and Console. The Console tab is active, displaying the output of the Java application: <terminated> Teste [Java Application] C:\Program Files\Java\jre-10\ 2018-05-13T18:15:37.703847700. The output is the ISO 8601 string representation of the current date and time.

Note que ao instanciar é utilizada uma função chamada *now()*, que faz com que sejam obtidos a data e hora atual do sistema.

É de extrema importância você ter esse objeto à disposição, pois ele é a base para se manipular a data ou a hora.

## LocalDate

Essa função tem como objetivo retornar apenas a data, retirando a hora, minutos e segundos do objeto gerado pelo **LocalDateTime**. Basicamente uma filtragem, vamos ao exemplo:

```
//Importando classes
import java.time.LocalDate;
import java.time.LocalDateTime;

public class Teste {

    public static void main(String[] args) {

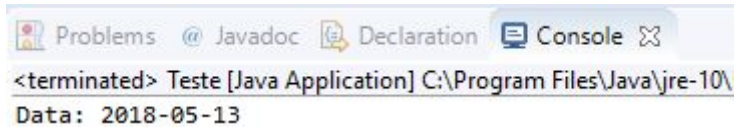
        //Instanciar objeto contendo a data e a hora
        LocalDateTime tempo = LocalDateTime.now();

        //Extraíndo apenas a data
        LocalDate data = tempo.toLocalDate();

        //Exibindo a data
        System.out.println("Data: " + data);

    }

}
```



The screenshot shows the bottom of a Java IDE window. At the top, there is a toolbar with icons for 'Problems', 'Javadoc', 'Declaration', and 'Console'. The 'Console' tab is active. Below the toolbar, the console output is displayed. It shows the text '<terminated> Teste [Java Application] C:\Program Files\Java\jre-10\' followed by a new line and the output 'Data: 2018-05-13'.

```
<terminated> Teste [Java Application] C:\Program Files\Java\jre-10\
Data: 2018-05-13
```

## LocalDate – Retornando hora, minuto e segundo

De maneira bem simples veja como podemos obter a hora completa:

```
//Importando classes
import java.time.LocalDateTime;

public class Teste {

    public static void main(String[] args) {

        //Instanciar objeto contendo a data e a hora
        LocalDateTime tempo = LocalDateTime.now();

        //Obtendo o hora, minuto e segundo
        int segundo = tempo.getSecond();
        int minuto = tempo.getMinute();
        int hora = tempo.getHour();

        System.out.println(hora+":"+minuto":"+segundo);

    }

}
```

Problems @ Javadoc Declaration Console

<terminated> Teste [Java Application] C:\Program Files\Java\jre-10\ 18:31:52

## LocalDate – Retornando dia, mês e ano

Se for seguir a mesma lógica do exemplo anterior para as datas teremos o seguinte:

```
1 //Importar classe
2 import java.time.LocalDateTime;
3
4 public class Principal {
5
6     public static void main(String[] args) {
7
8         //Instanciando um objeto contendo a data e a hora
9         LocalDateTime tempo = LocalDateTime.now();
10
11         //Obtendo o dia, mês e ano
12         int dia = tempo.getDayOfMonth();
13         int mes = tempo.getMonthValue();
14         int ano = tempo.getYear();
15
16         //Exibir data
17         System.out.println(dia+"/"+mes+"/"+ano);
18
19     }
20
21 }
```



Console

<terminated> Principal (21) [Java Application] C:\Program Files\Java\jre1.8.0\_152\l  
24/5/2018

Para maiores informações sobre essa classe você poderá acessar o site completo sobre ela:

<https://docs.oracle.com/javase/8/docs/api/java/time/LocalDateTime.html>

Fonte para a elaboração deste tutorial:

- [https://www.tutorialspoint.com/java8/java8\\_datetime\\_api.htm](https://www.tutorialspoint.com/java8/java8_datetime_api.htm)
- <http://blog.caelum.com.br/conheca-a-nova-api-de-datas-do-java-8>
- <http://www.botecodigital.info/java/manipulando-datas-em-java> (Versão 7 ou inferior)

## Exercícios

1. Faça com que seja exibida a data e hora em formato pt-br, exemplo:  
14/05/2018 – 13:58

E abaixo escrito por extenso:

Hoje é dia quatorze de maio de dois mil e dezoito.

A hora atual é treze horas e cinquenta e oito minutos.

2. Criar um sistema para cadastrar os seguintes dados:
  - a. Nome do produto
  - b. Valor
  - c. Quantidade em estoque

Armazene em um ArrayList esses dados, e juntamente com eles adicione a data e a hora que foram cadastrados.

O sistema além da opção de cadastro deverá ter a opção listar para exibir os dados dos produtos informados (nome, valor, quantidade em estoque, data e hora).