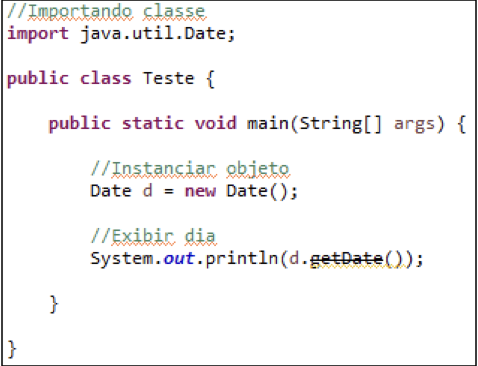
**Classe Date**

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:

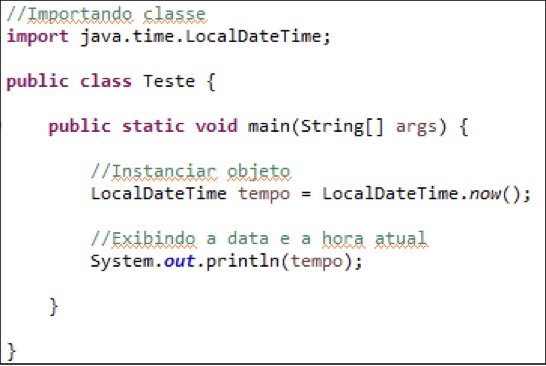


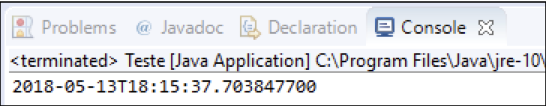
Note que na imagem acima foi instanciado um objeto da classe **Date** e quando chamado o método **getDate()** ele está com um risco. Sempre que você notar esse risco o Java considera esse comando *deprecated*ou em sua tradução comando em desuso.

A classe **Date** é muito comentada, porém a 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 porventura 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, veja abaixo a estrutura básica retornando data e hora completas:

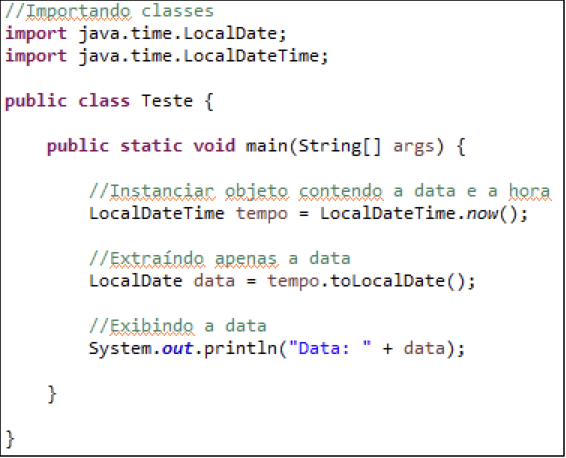
****

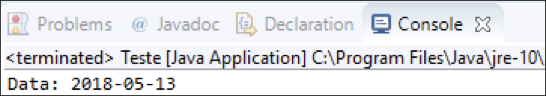
****

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**

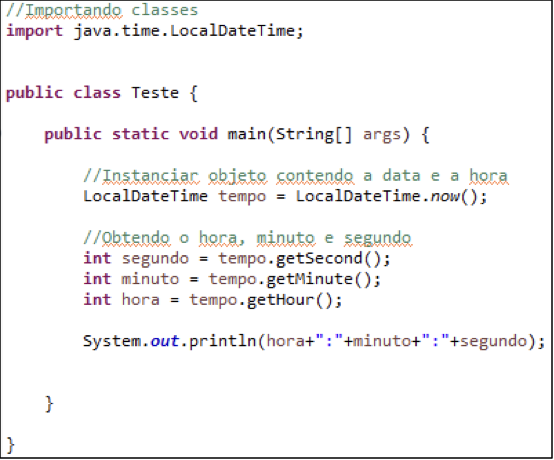
Essa função tem como objetivo retornar apenas a data, retirando a hora, minutos e segundos do objeto gerado pelo **LocalDateTime**. Basicamente uma filtragem, vamos ao exemplo:

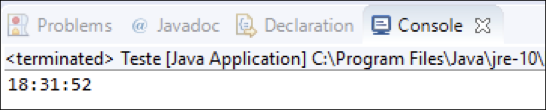


****

**Retornando hora, minutos e segundos**

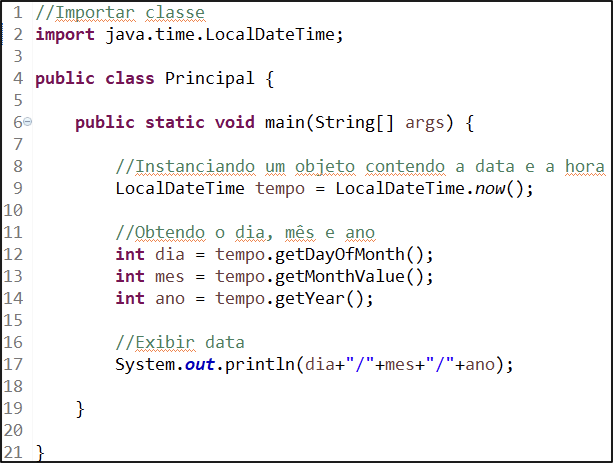
De maneira bem simples veja como podemos obter a hora completa:

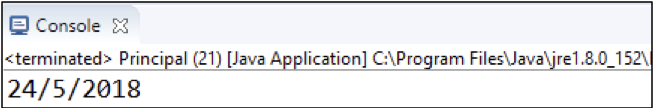




**Retornando dia, mês e ano**

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





Fontes para a elaboração deste tutorial:

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

**Atividades**

1. Exiba a data e a hora no formato pt-br, exemplo: 02/10/2022 - 14:32, além disso deverá retornar por extenso a data e a hora.
2. Criar um sistema para cadastrar os seguintes dados:

* Nome do produto
* Valor
* 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 para exibir os dados dos produtos informados (nome, valor, quantidade em estoque, data e hora).

1. Desenvolva um sistema onde irá gerar um número entre 0 e 100. O usuário deverá tentar acertar esse número, enquanto não acertar o número gerado deverá ser perguntado um número para realizar o comparativo.

Sempre que o palpite dado pelo usuário for diferente do número gerado deverá ser informado os seguintes auxílios:

* Se a diferença for até 10 retorne: Está perto.
* Se a diferença for entre 11 e 20 retorne: Está longe.
* Diferença acima de 20 retorne: Muito longe.

Após acertar o número gerado, retorne a quantidade de tentativas realizadas e uma das frases abaixo:

* Caso acerte de primeira: "Nossa! Acertou na mosca"
* Até 5 palpites - "Parabéns! Você é bom de adivinhação"
* Até 10 - "Muito bom, continue assim"
* Até 20 - "Podia ser melhor, continue tentando"
* Acima de 20 - "Meio lerdo você, hein?"