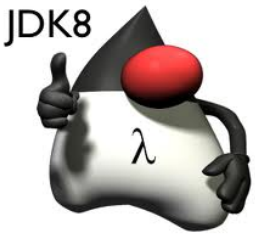
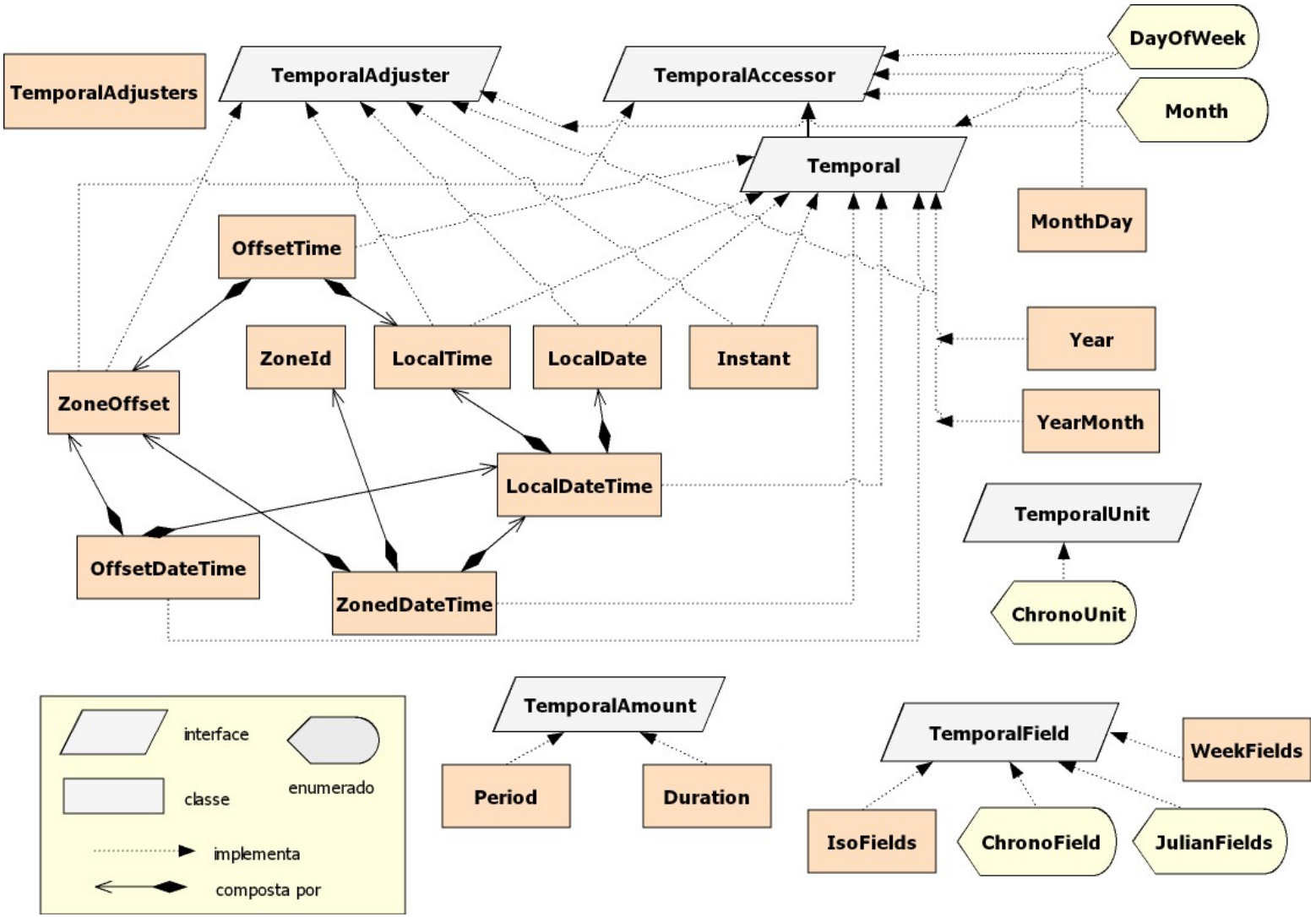


EXERCÍCIOS PRÁTICOS I

Date Time API: Exercícios

Tenha sempre em atenção a arquitectura da API e a compatibilidade de tipos.



Date Time API: Exercícios

Realize os exercícios criando uma pequena classe na qual vai inserindo uma descrição da questão a resolver e o código a testar (colocando em comentário /* */ o código já testado e seus resultados se o entender), por exemplo como em:

```
import java.time.LocalDate;
import java.time.Month;
import java.time.Period;

public class Testes_TimeDateAPI {
    public static void main(String[] args) {

        // 1.- Cálculo da Idade de uma pessoa
        LocalDate hoje = LocalDate.now();
        System.out.println("Hoje : " + hoje);
        LocalDate dataNascim = LocalDate.of(1986, Month.APRIL, 26);
        System.out.println("Data Nascim : " + dataNascim);
        Period p = Period.between(dataNascim, hoje);
        System.out.printf("Idade : %d Anos %d Meses %d Dias %n",
                          p.getYears(), p.getMonths(), p.getDays());
    }
}
```

Date Time API: Exercícios

☞ Adicionalmente, e para todos os exemplos com algum interesse geral, considere colocar o seu código como sendo um método de classe de uma classe utilitária do tipo **Utils_DateTime**, cf.

```
public class Utils_DateTime {

    // Cálculo da Idade actual em anos
    public static int idadeHoje(LocalDate dataNascim) {
       LocalDate hoje = LocalDate.now();
       LocalDate dataNascim = LocalDate.of(1986, Month.APRIL, 26);
       return Period.between(dataNascim, hoje).getYears();
    }
  ……
}
```

☞ Sempre que possível generalizar o código procurando usar interfaces como parâmetros de entrada ou de saída.

Date Time API: Exercícios

1.- Crie uma **LocalDateTime** com a sua data de nascimento e converta-a para um **Instant**. Realize a conversão contrária e teste o rigor da conversão inversa.

2.- Crie uma **Duration** de 1 hora usando o construtor próprio e uma outra usando a adequada **ChronoUnit**. Mostre que são iguais em minutos, segundos e nanosegundos.

3.- Quando eu nasci, o meu pai tinha 33 anos e 6 meses e a minha mãe 31 anos e 9 meses. Que idade temos os três agora ? A minha irmã nasceu 3 anos, 1 mês e 25 dias depois de mim. Em que data nasceu e que idade tem agora ?

4.- Determine a **Duration** e o **Period** diferença entre duas **LocalDateTime**. Determine a lista de todas as unidades temporais contidas na duração resultado. Verifique com a máxima precisão que as diferenças de tempo são iguais.

5.- Considere o dia de hoje. Verifique a que Era, Milénio e Século correspondente.

6.- Calcule todas as possíveis **ChronoUnit** de diferença entre duas **LocalDateTime**. Escolha em seguida a data-tempo mais antiga e verifique se ao somar-lhe os valores das crono-unidades se encontra a outra data-tempo.

Date Time API: Exercícios

7.- O seguro do meu carro termina a AAAA-MM-DD. Quantas semanas faltam? E quantos dias ? Apresentar mais do que uma solução.

8.- O seguro do meu carro termina a AAAA-MM-DD. Quero renová-lo logo no primeiro dia dessa semana. Qual é essa data ? Apresentar mais do que uma solução.

9.- Determine a **LocalTime** actual e usando todos os **ChronoField** verifique se fazem parte de tal data-tempo. Considerando apenas **ChronoField. MINUTE_OF_DAY** determine o seu valor, o nome da **ChronoUnit** correspondente, a gama de valores possíveis e se é um campo que integra uma data.

10.- Determine o **Instant** actual no nosso fuso. Converta-o para a nossa actual **ZonedDateTime** e para a **ZonedDateTime** da Austrália. A partir desta última ZDT calcule o respectivo **Instant** e compare-o com o do nosso fuso anteriormente calculado.

11.- Converter instâncias locais (verificar se o são) de **Date** para **LocalDateTime** e instâncias de **GregorianCalendar** para **ZonedDateTime**.

12.- Pago o meu seguro automóvel na modalidade trimestral. Fiz o 1º pagamento a 30-09-2017. Quais as datas dos pagamentos seguintes ?

Date Time API: Exercícios

13.- Dado um número real correspondente a um valor em dias inferior a 366, convertê-lo para dias, horas, minutos e segundos (usando uma **LocalDateTime** ?). Uma **lunação** (período sinótico da lua correspondente a 4 fases seguidas) demora 29,530589 dias, e entre duas fases passam-se em média 7,38 dias. Calcule estes dois valores em dias, horas, minutos, segundos e milisegundos usando o código anterior.

14.- Escreva código que converta uma qualquer **Duration** numa **LocalDateTime**.

15.- No dia 31/12/2017 quero enviar votos de Bom Ano a um grupo de pessoas que estão em Santiago do Chile. Quero fazê-lo meia hora antes da passagem de ano deles. A que horas de Braga o devo fazer ? E se eu estivesse em Macau ?

16.- Vou ao ginásio todas as 4ªs feiras. Pago 5 euros por sessão. Quanto vou pagar até ao fim do ano ?

17.- Determine quais os anos bissextos dos séculos XX e XXI.

18.- Verifique se para um dado ano e data existe diferença entre o número da semana actual e o número de semanas para o fim do ano, num calendário normal e num week-based-year.

Date Time API: Exercícios

19.- Como determinar a data do 3º dia da 2ª semana de Dezembro de um ano dado ?

20.- Listar todas as **ZoneId** com offsets que não têm horas certas.

21.- Em que dia da semana foi o 25 de Abril de 1974. Quantos anos e dias passaram?

22.- Para um dado ano determine as datas de todos os domingos.

23.- Dada uma **LocalDate** determine o valor da data correspondente a mais 10 dias úteis.

24.- Fala-se muito de um dia especial, o "meio do ano". Para um ano dado, determine a data que melhor pode representar o meio desse ano.

25.- Dada uma factura emitida num dado dia de um mês do um ano, de termine as datas correspondentes ao seu pagamento a 30, 60 e 90 dias.

26.- Importe a classe **java.time.chrono.MinguoDate** correspondente ao calendário chinês usado, por exemplo em Taiwan, sendo que 0001-01-01 (Minguo) corresponde a 1912-01-01 (ISO) . Confirme esta correspondencia de datas e determine a data-tempo actual no calendário Minguo.

Date Time API: Exercícios

27.- Considere os **TemporalAdjuster** firstDayOfMonth() e lastDayOfMonth() e verifique se para uma dada data se verifica que o último dia do mês anterior da data é o dia anterior ao primeiro dia do mês da data.

28.- Defina um **TemporalAdjuster** de nome **quinzeDiasAntes(Temporal t)** que ajuste uma data para exactamente duas semanas antes. Teste-o com a data de hoje.

29.- Dadas duas **ZoneId** calcular a diferença de tempo entre as data-tempo locais numa dada ChronoUnit .

30.- Crie um **Comparator<LocalDate>** e teste-o na ordenação de um **TreeSet<LocalDate>**. Use no seu exemplo código utilitário de Java tal como **Arrays.asList()** para criar uma lista com as datas a ordenar. Teste igualmente o comparador usando o utilitário **Collections.sort()**.

Date Time API: Exercícios

31.- Dado um ficheiro de texto **transCaixa.txt** contendo em cada linha o registo de uma transacção de caixa datada, no formato Nº de Transacção (Tddd), Nº de Caixa (dd), Valor em Euros (ddd.dd) e data (cf. DD:MM:AAAAThh:mm), separados por "/", todas do ano de 2017, cf. os exemplos

T148/29/37.28/6:11:2017T19:28
T951/9/30.6/5:10:2017T9:24
T216/7/69.69/1:1:2017T15:50

Crie uma classe **TansCaixa** que represente cada uma destas transacções através dos atributos Nº de transacção, Nº de Caixa, Valor e Data e Hora local (**LocalDateTime**); Use métodos de classe **of()** para criar as instâncias;

Crie um contentor adequado e leia as transacções de **transCaixa.txt** para esse contentor. Use o seguinte código utilitário de JAVA8:

```
List<String> lines = null;
try { lines = Files.readAllLines(Paths.get(nomeFich)); }        //StandardCharsets.UTF_8
catch(IOException exc) { System.out.println(exc.getMessage()); }
  ……..
```

Date Time API: Exercícios

Programe em seguida as seguintes consultas:

- **Nº total de transacções e valor total de facturação;**
- **Nº de transacções realizadas entre as 8H00 e as 20H00 em 3 datas válidas indicadas;**
- **Tabela com nº total de transacções por dia da semana;**
- **Comparar a facturação total das SEXTAS com a dos SÁBADOS;**
- **Comparar a facturação por semanas do mês (1 a 4/5);**
- **outras**