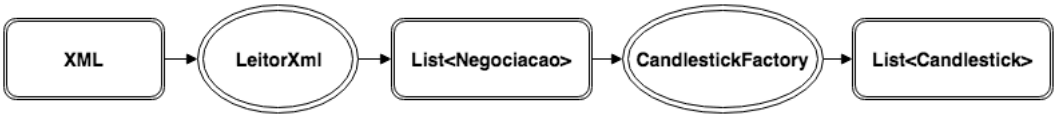
09

Colocando em prática: constroiCandles

Chegou a hora de implementar o método que separa os candles. Esse novo método será criado na classe `CandlestickFactory` e recebe a `List<Negociacao>` que vem do nosso `LeitorXml` :



Mãos à obra, mas começando pelos testes!

1) Vamos adicionar o método `paraNegociacoesDeTresDiasDistintosGeraTresCandles` na classe `CandlestickFactoryTest` . Ele vai criar diversas negociações de três dias diferentes e deve nos retornar três candlesticks distintos. Como o código é mais complexo vamos passá-lo por completo:

```
@Test
public void paraNegociacoesDeTresDiasDistintosGeraTresCandles() {

}
```

2) Dentro desse método crie negociações de data diferentes:

```
@Test
public void paraNegociacoesDeTresDiasDistintosGeraTresCandles() {

    LocalDateTime hoje = LocalDateTime.now();

    Negociacao negociacao1 = new Negociacao(40.5, 100, hoje);
    Negociacao negociacao2 = new Negociacao(45.0, 100, hoje);
    Negociacao negociacao3 = new Negociacao(39.8, 100, hoje);
    Negociacao negociacao4 = new Negociacao(42.3, 100, hoje);

    LocalDateTime amanha = hoje.plusDays(1);

    Negociacao negociacao5 = new Negociacao(48.8, 100, amanha);
    Negociacao negociacao6 = new Negociacao(49.3, 100, amanha);

    LocalDateTime depois = hoje.plusDays(2);

    Negociacao negociacao7 = new Negociacao(51.8, 100, depois);
    Negociacao negociacao8 = new Negociacao(52.3, 100, depois);

    List<Negociacao> negociacoes = Arrays.asList(negociacao1, negociacao2, negociacao3, negocia
    negociacao5, negociacao6, negociacao7, negociacao8);

    //aqui vem mais
}
```

3) Depois da lista de negociações crie a fábrica de candles e chame o método `constroiCandles` (que ainda não existe):

```
@Test
public void paraNegociacoesDeTresDiasDistintosGeraTresCandles() {

    //criacao das negociacoes omitida

    CandlestickFactory fabrica = new CandlestickFactory();

    List<Candlestick> candles = fabrica.constroiCandles(negociacoes);

    //aqui vem mais
}
```

4) Agora vem os `Assert`, verificando os candles da lista:

```
@Test
public void paraNegociacoesDeTresDiasDistintosGeraTresCandles() {

    //criacao das negociacoes omitida

    //construindo candles

    Assert.assertEquals(3, candles.size());
    Assert.assertEquals(40.5, candles.get(0).getAbertura(), 0.00001);
    Assert.assertEquals(42.3, candles.get(0).getFechamento(), 0.00001);
    Assert.assertEquals(48.8, candles.get(1).getAbertura(), 0.00001);
    Assert.assertEquals(49.3, candles.get(1).getFechamento(), 0.00001);
    Assert.assertEquals(51.8, candles.get(2).getAbertura(), 0.00001);
    Assert.assertEquals(52.3, candles.get(2).getFechamento(), 0.00001);
}
```

5) Por fim, implemente o método `constroiCandles` na classe `CandlestickFactory`. Temos a seguinte implementação, o importante é que você entenda a lógica que separa as candles e, claro, faça que os testes passam:

```
public List<Candlestick> constroiCandles(List<Negociacao> negociacoes) {

    // Cria a lista de negociações do dia e a lista de Candles que devemos retornar
    List<Candlestick> candlesticks = new ArrayList<>();

    List<Negociacao> negociacoesDoDia = new ArrayList<>();

    LocalDateTime dataAtual = negociacoes.get(0).getData();

    // Vamos percorrendo a lista com todas as negociacoes.
    for (Negociacao negociacao : negociacoes) {
        // se for mesmo dia, adiciona na lista de Negociacoes daquele dia
        if(negociacao.isMesmoDia(dataAtual)){
            negociacoesDoDia.add(negociacao);
        }else{
            // se não for mesmo dia, fecha o candle e reinicia variáveis
            Candlestick candle = geraCandleParaData(negociacoesDoDia, dataAtual);
            candlesticks.add(candle);


            negociacoesDoDia = new ArrayList<>();
```

```
            // Aqui precisamos adicionar o item que caiu no else na nova lista, caso constrário
            negociacoesDoDia.add(negociacao);

            dataAtual = negociacao.getData();
        }

    }
    // adiciona último candle
    Candlestick candle = geraCandleParaData(negociacoesDoDia, dataAtual);
    candlesticks.add(candle);

    return candlesticks;
}
```

6) Execute os testes da classe `CandlestickFactoryTeste` !