

# Tarefa avaliada por colega: Quebra de Strings com CamelCase

## Você foi aprovado!

Parabéns. Você recebeu 95/100 pontos. Analise seu feedback abaixo e continue o curso quando estiver pronto. Você também pode ajudar mais colegas avaliando seus envios.

i

Parece que esta é sua primeira tarefa com avaliação entre colegas.

1. [Instruções](#)
2. [Meus envios](#)
3. [Discussões](#)

Criar utilizando TDD um método que transforma uma cadeia de caracteres em camel case (<http://pt.wikipedia.org/wiki/CamelCase>) em uma lista de Strings com as palavras. O método deve possuir a seguinte assinatura: "public static List<String> converterCamelCase(String original)".

Faça um relatório detalhando todo o processo de TDD.

Para cada ciclo que você fizer no TDD, adicione uma seção no relatório o seguinte: o teste adicionado, como estava o código antes, como ficou o código depois para fazer o teste passar e uma pequena descrição textual do que foi feito.

Abaixo seguem alguns exemplos de entrada e saída que você pode usar como base para os seus testes (crie adicionais ou diferentes se achar necessário):

- nome - “nome”
- Nome - “nome”
- nomeComposto - “nome”, “composto”
- NomeComposto - “nome”, “composto”
- CPF - “CPF”
- numeroCPF - “numero”, “CPF”
- numeroCPFContribuinte - “numero”, “CPF”, “contribuinte”
- recupera10Primeiros - “recupera”, “10”, “primeiros”
- 10Primeiros - Inválido → não deve começar com números
- nome#Composto - Inválido → caracteres especiais não são permitidos, somente letras e números

É permitida a criação de métodos auxiliares. Para ficar mais divertido e praticar a refatoração, nenhum método pode possuir mais de dez linhas de código em seu corpo. Não vale “roubar” e incluir vários comandos em uma mesma linha de código!

Para cada refatoração que precisar fazer nesse processo, adicione uma seção no relatório o seguinte: como estava o código antes, como ficou o código depois da refatoração e uma pequena descrição textual do que foi feito.

No método desenvolvido é permitida somente a utilização de classes da API básica da linguagem Java. Se você utilizar algum componente externo que facilite demais sua tarefa, estará tirando o propósito do exercício!

Você deverá entregar o relatório detalhado e o código final.

Review criteria

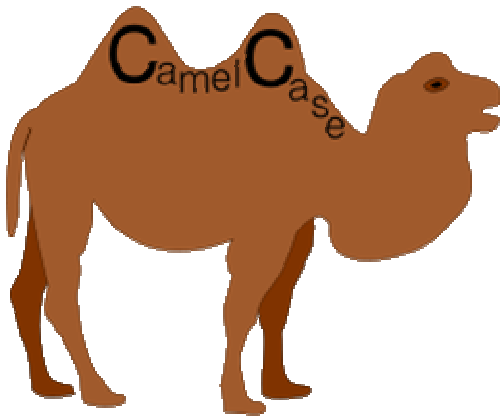
Você será avaliado com base no seguinte:

- Cumprimento dos requisitos de implementação pedidos no enunciado
- Organização do código implementado (seguindo a restrição de no máximo 10 linhas de código por método)
- A utilização de TDD no processo de desenvolvimento

CamelCase e Refatoração

### **CamelCase**

CamelCase é a denominação em inglês para a prática de escrever palavras compostas ou frases, onde cada palavra é iniciada com Maiúscula, com exceção às vezes da primeira letra da primeira palavra, e unidas sem espaços. É um padrão largamente utilizado em diversas linguagens de programação, como Java, C#, Ruby, PHP e Python, principalmente nas definições de Classes e Objetos. Pela sua associação com tecnologia, o marketing se apropriou dessa maneira de escrever, injetando certo ar de "tecnologia" nos produtos assim nomeados: iPod, GameCube, OpenOffice.org, StarCraft, dentre outros.



A provável origem do termo é a semelhança do contorno de expressões CamelCase, onde as letras em maiúsculo "saltam" no meio das minúsculas como corcovas de um camelo.

Fonte: <https://pt.wikipedia.org/wiki/CamelCase>

### **Refatoração**

É uma técnica para transformar um programa com problemas de projeto e codificação, como código duplicado e nomes inapropriados de métodos e classes, por exemplo,

eliminando tais problemas. Neste curso, iremos mostrar uma maneira segura de fazer refatoração ou transformação do código sem que o comportamento anterior à refatoração seja mudado. O que garante que o código transformado não modificou o comportamento é testar o código antes e depois da refatoração e ele deve passar com sucesso em ambos os testes.

Neste trabalho, não se espera que você faça uso de técnicas avançadas e testadas de refatoração, pois na Semana 1 só mostramos um pouco dos conceitos do assunto. O que se deseja é que, por causa do limite de linhas de código do método, você realize eliminações apropriadas de código redundante, garantindo que o comportamento não mude com as alterações que você promoveu. Para isso, use a técnica de testar antes e depois de fazer suas refatorações. Use também a Regra de Três para só iniciar a refatoração quando a redundância ocorre pela terceira vez, a menos que o número de linhas já ultrapasse o limite de 10 linhas colocado na atividade; nesse caso, a refatoração tem que ser feita imediatamente.