coursera

ITA - TDD - Week 2

Desenvolvimento de Software Guiado por Testes

Instruções do desafio:

No Curso 1, nós projetamos um conjunto de classes, usando a Modelagem CRC, para o SAB – Sistema de Automação de Biblioteca. A maioria dos métodos implementados precisa de refatoração, que não foi feita para podermos usar algum deles como exercício de revisão por pares.

Assim solicitamos que examine o método abaixo registraUsuario(String) e faça o seguinte:

a) Identifique uma lista de maus cheiros que você encontra no código, relacionando cada um deles com o correspondente tipo de mau cheiro exercitado nesta parte do curso: [mau cheiro no código (trecho do código)/tipo de mau cheiro (de acordo com Fowler, pode estar em português)].

b) Realize o Ciclo de Refatoração apresentado, eliminando cada um dos maus cheiros encontrado no código do método, considerando apenas os tipos de mau cheiro exercitados nesta Semana 2 do curso.

c) Entregue um documento em que você apresenta o seguinte:

A) Código anterior do método registraUsuario(String), antes de iniciar o Ciclo de Refatoração.

B) Imagem: Imagem da execução bem-sucedida (verde) no Eclipse ou outro ambiente Java, comprovando que código atual do SAB, incluindo o método registraUsuario(String) está funcionando direito (pelo menos de acordo com a bateria de testes atual).

C) Ciclo de Refatoração até a Lista de Maus Cheiros ficar vazia, apresentando 5 coisas para cada refatoração realizada no Ciclo de Refatoração:

1. **Antes**: O código Antes da refatoração, com o trecho a ser refatorado com as letras coloridas ou com fundo amarelo
2. **Tipo Mau Cheiro/Técnica de Refatoração**: Indique o tipo do mau cheiro identificado no código em 1) e a técnica de refatoração empregada, ambos de acordo com Fowler e podendo estar em português!
3. **Depois**: O código Depois da refatoração, com o trecho refatorado com as letras coloridas ou com fundo verde

4. **Imagem**: Imagem da execução bem-sucedida (verde) no Eclipse ou outro ambiente Java, comprovando que a refatoração foi feita a contento!
5. **Lista de maus cheiros**: Atualize a lista, eliminando o mau cheiro que deu origem à refatoração deste ciclo

D) Código Depois final do método abaixo registraUsuario(String), sem letras ou fundo coloridos!

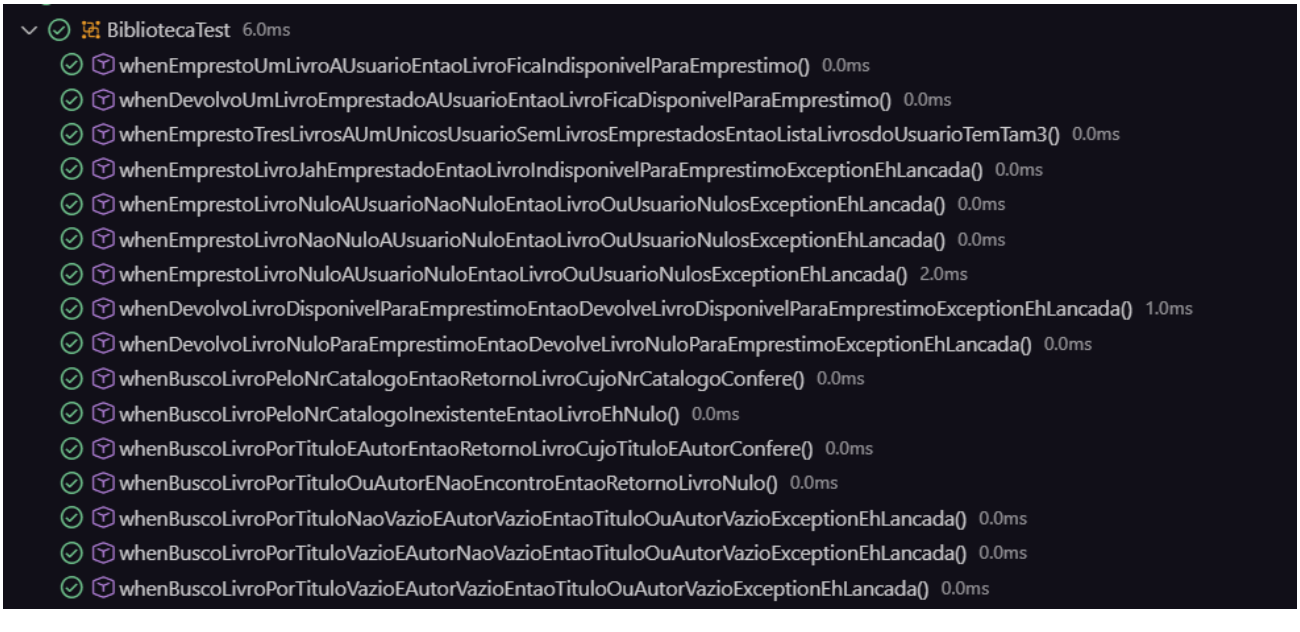
Procure apresentar o código sempre bem apresentado, de acordo com boas práticas de apresentação/formatação de código Java. Pode usar, por exemplo, o Source/Format do Eclipse ou equivalente do seu ambiente Java.

A bateria de testes atual não pode ser modificada de forma alguma durante o Ciclo de Refatoração!

```
public void registraUsuario(String nome)
        throws UsuarioJaRegistradoException, UsuarioComNomeVazioException,
        UsuarioInexistenteException {
    if (nome != null) {
        if (!nome.isEmpty()) {
            Usuario usuario = new Usuario(nome);
            if (!_usuarios.contains(usuario)) {
                _usuarios.add(usuario);
            } else
                throw new UsuarioJaRegistradoException("--->Já existe usuário com o nome \""
                        + nome + "\"! Use outro nome!");
        } else
            throw new UsuarioComNomeVazioException("--->Não pode registrar usuario com nome vazio!");
    } else
        throw new UsuarioInexistenteException("--->Não pode registrar usuario inexistente!");
}
```

Refatoração de acordo com os requisitos do desafio:

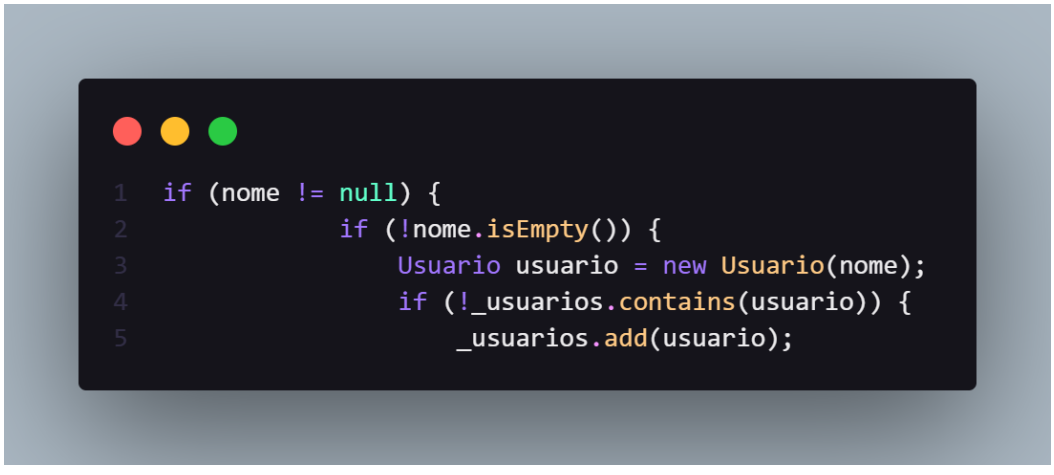
A imagem abaixo comprova que antes de iniciar a refatoração, todos os testes propostos pelo modelo disponibilizado para desenvolvimento do desafio, estão passando!!



1- Após garantirmos que os testes estão ok, analisamos o código e foram encontrados os seguintes Code Smells a nível de design de código que podem ser categorizados como Object-

Orientation Abusers que na sua definição"Trata-se da violação à orientação a objetos. Assim a aplicação inadequada total ou parcialmente da herança, polimorfismo ou encapsulamento podem provocar problemas de código. Essa categoria de Code Smells pode ser indentificada em códigos que possuem muitos ifs aninhados ou switches complexos", segue abaixo o código dos Code Smells identificados no método:

1.1- Condições negativas:



```
if (nome != null) {
            if (!nome.isEmpty()) {
                Usuario usuario = new Usuario(nome);
                if (!_usuarios.contains(usuario)) {
                    _usuarios.add(usuario);
```

1.1- Ifs aninhados:

```
if (nome != null) {
            if (!nome.isEmpty()) {
                Usuario usuario = new Usuario(nome);
                if (!_usuarios.contains(usuario)) {
                    _usuarios.add(usuario);
                } else
                    throw new UsuarioJaRegistradoException("--->Ja existe usuario com o nome \""
                            + nome + "\"! Use outro nome!");
            } else
                throw new UsuarioComNomeVazioException("--->Nao pode registrar usuario com nome vazio!");
        } else
            throw new UsuarioInexistenteException("--->Nao pode registrar usuario inexistente!");
```

2- Com as informações adquiridas na etapa anterior, partimos para a refatoração do primeiro caso de Code Smells - condições negativas e ifs aninhados, a refatoração objetivou tornar as condições positivas aplicando a técnica de Early Return, mantendo as funcionalidades e regras do sistema intáctas, ou seja ao final do ciclo de refatoração todos os testes devem continuar passando em suas validações:

```
public void registraUsuario(String nome)
        throws UsuarioJaRegistradoException, UsuarioComNomeVazioException,
        UsuarioInexistenteException {
    if (nome == null) throw new UsuarioInexistenteException("--->Nao pode registrar usuario inexistente!");
    if (nome.isEmpty()) throw new UsuarioComNomeVazioException("--->Nao pode registrar usuario com nome vazio!");
    Usuario usuario = new Usuario(nome);
    if (_usuarios.contains(usuario)) throw new UsuarioJaRegistradoException("--->Ja existe usuario com o nome \"" + nome + "\"! Use outro nome!");
    _usuarios.add(usuario);
}
```

1.1- Após a etapa de refatoração, foi feita a etapa de validação dos testes, afim de garantir que os requisitos e funcionalidades do sistema não foram afetadas pela etapa

de refatoração: