Aridio Gomes da Silva - aridiosilva@aridiosilva.com - https://www.linkedin.com/in/aridio-silva-74997111/

I - INTRODUÇÃO

Este é o meu **Relatório** da **TAREFA da SEMANA DOIS** do **Curso de Desenvolvimento Direcionado à Testes (TDD) da Plataforma Coursera em parceria com o ITA,** contendo ***descrição e detalhamento solicitados***, bem como atendendo às ***recomendações e requisitos*** especificados no documento "*Tarefa avaliada por colega: Refatoração do SAB*".

A seguir as seções deste relatório seguem o formato e sequência solicitada, e que são respectivamente:

II – Lista de Maus Cheiros Identificados no Método *registraUsuario(String)*

Considerando que, foi estabelecido como obrigatório para esta tarefa, que somente fossem considerados os maus cheiros estudados na Semana Dois do Curso Coursera/ITA, e mesmo havendo a presença de outros tipos de bad smells no método exigido para análise e refatoração, informo que, no Código do Método ***registraUsuário(string)*** constam os seguintes tipos de maus cheiros e o e o respectivo número de ocorrências associadas a cada tipo, respectivamente:

#	Tipo do Mau Cheiro Encontrado no Código	Nº de Ocorrências Identificadas
1	*Ifs com Expressões Booleanas Negativas*	*3 (três)*
2	*Ifs Aninhados*	*3 (três)*
3	*Código Duplicado*	*2 (duas)*

III – Localização dos Maus Cheiros no Código do Método *registraUsuario(String)*

➢ Mau Cheiro UM : Ifs com Expressões Booleanas Negativas

O operador de negação (***Not***) é um operador lógico, representado em Java pelo ***símbolo "!"*** (*exclamação*). Este operador de negação inverte (ou nega) o valor do seu operando. O uso do operador de negação, segundo a literatura técnica, compromete a legibilidade do nosso código e dificulta a compreensão. Assim, condicionais negativas são muito mais difíceis de entender do que as positivas, sendo, portanto, considerada uma prática enquadrada como *bad smell* no nosso código, e que deve ser removida.

O ***Livro Clean Code*** apresenta na página 302, da edição de 2009, a seção intitulada "***G29 Evitar Condicionais Negativas***", parte integrante do ***Capítulo 17 - Bad Smells***, onde o autor afirma que: "*as condicionais negativas devem ser evitadas, pois são mais difíceis para entender do que as positivas*". Assim, recomenda "*sempre que for possível, substitui-las por expressões positivas*".

Abaixo, o código de produção do *Método **registraUsuário** do Sistema SAB*, fornecido originalmente para esta tarefa de refatoração, onde aparecem evidenciados na cor amarela os locais onde constam as três ocorrências de Expressões Booleanas Negativas, respectivamente:

```
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 usuário com nome vazio!");
   } else
       throw new UsuarioInexistenteException("--->Não pode registrar usuário inexistente!");
}
```

Expressões booleanas negativas

Aridio Gomes da Silva - aridiosilva@aridiosilva.com - https://www.linkedin.com/in/aridio-silva-74997111/

➢ Mau Cheiro DOIS : Ifs Aninhados

Em programação orientada à objetos o *uso inadequado de estruturas do tipo IF-THEN-ELSE e o seu aninhamento* são considerados como um *mau cheiro no código*, visto que, dependendo da *profundidade do aninhamento* de IF-THEN-ELSEs, aumenta consideravelmente a *complexidade do código*, diminui a *clareza e legibilidade*, e dificulta o *entendimento* do que faz o código, além de *aumentar o tamanho do código do método em número de linhas.*

Abaixo, o código de produção do *Método **registraUsuário** do Sistema SAB*, fornecido originalmente para esta tarefa de refatoração, possui três ocorrências de aninhamento de IF-THEN-ELSE conforme mostrado hachuriado em cor amarela no código abaixo:

```
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 usuário com nome vazio!");
    } else
        throw new UsuarioInexistenteException("--->Não pode registrar usuário inexistente!");
}
```

Ninhos de IF-THEN-ELSE

➢ Mau Cheiro TRÊS : Código Duplicado

Uma das regras mais importantes enfatizadas no *Livro Clean Code*, é que devemos considerar seriamente em não desrespeitar o *Bom Princípio de Projeto* denominado *DRY (Don´t Repeat Yoursef).* Este princípio é, também, um dos principais na *Programação Extrema (XP)*, onde o seu criador, *Kent Beck*, também *pai do TDD*, define como "*Once, and only once*" - Isto é, "*escreva uma vez e somente uma vez*". Cada duplicação no código represente uma oportunidade perdida de abstrair. Este tema de duplicação de código é o mau cheiro mais falado, sendo considerado pelo *Martin Fowler*, *pai da Refatoração*, como *o número um dos maus cheiros*.

A duplicação de código está presente no *Método **registraUsuário** do Sistema SAB*, fornecido originalmente para esta tarefa de refatoração, e possui três ocorrências de ELSE, conforme mostrado no hachuriado em cor amarela no código abaixo. À medida que, as três ocorrências de cada dum dos maus cheiros 1 e 2 forem eliminadas, três dos ELSES *ficarão inúteis* e serão *código morto*, como pode ser visto mais a frente neste documento.

CÓDIGO DUPLICADO

```
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 usuário com nome vazio!");
    } else
        throw new UsuarioInexistenteException("--->Não pode registrar usuário inexistente!");
}
```

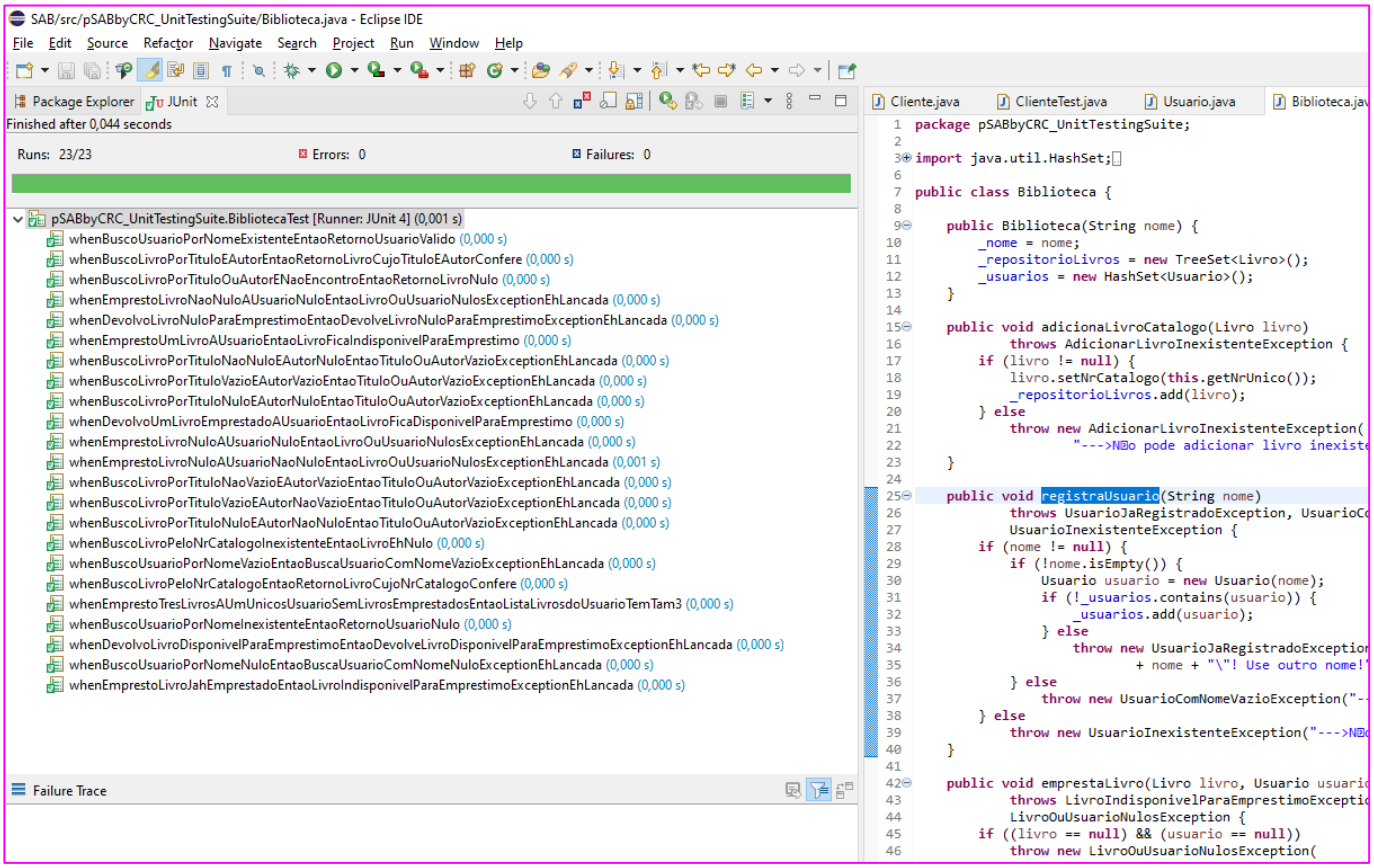
Aridio Gomes da Silva - aridiosilva@aridiosilva.com - https://www.linkedin.com/in/aridio-silva-74997111/

III – Situação do Código Antes de Refatorar

```
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("--->J existe usu rio com o nome \""
                        + nome + "\"! Use outro nome!");
        } else
            throw new UsuarioComNomeVazioException("--->N o pode registrar usuario com nome vazio!");
    } else
        throw new UsuarioInexistenteException("--->N o pode registrar usuario inexistente!");
}
```

IV – Imagem da *Execução Bem-Sucedida* (verde) do *Código de Produção* e *Bateria de Testes* no Eclipse

Abaixo segue a Imagem da execução bem-sucedida (verde) no *Eclipse IDE Versão 2020-09*, comprovando que ***código atual do SAB***, incluindo o método ***registraUsuario(String)*** está funcionando direito (e de acordo com a bateria de testes provida pelo ITA).



Para processamento da Bateria de Casos de Testes Unitários providos no projeto SAB, foi utilizado o JUnit Versão 4 e o JRE com a versão 8, e utilizado o ECLIPSE IDE (Integrated Development Environment) Versão 2020-09 para realizar as refatorações, compilações e execução dos testes unitários.

Aridio Gomes da Silva - aridiosilva@aridiosilva.com - https://www.linkedin.com/in/aridio-silva-74997111/

V – O Ciclo da Refatoração do Código do Método registraUsuario(String)

➢ Ciclo 1 da Refatoração – Ocorrência 1 do Mau Cheiro 1 - Ifs com Expressões Booleanas Negativas

- **Antes de Refatorar:**

```
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 usuário com nome vazio!");
    } else
        throw new UsuarioInexistenteException("--->Não pode registrar usuário inexistente!");
}
```

- **Fragmento do Código a Refatorar neste Ciclo 1:**

```
    if (nome != null) {
```

- **Tipos de Técnicas de Refatoração Usadas**

A refatoração para ***eliminação da primeira ocorrência do mau cheiro 1 (If com Expressão Booleana Negativa)*** a ser realizado envolve a adoção das duas técnicas, respectivamente:

- *Trocar as condicionais Negativas para Positivas, uma por vez e*
- *Trocar o bloco de código associado do ELSE para o lugar do bloco de código do IF e vice-versa*

Assim sendo, terei que realizar dois passos neste ciclo 1, de modo que não seja alterado o comportamento do código e que possa compilar sem erros, e rodar a bateria de testes de modo que continuem passando. Portanto, procedi o seguinte no processo da refatoração neste ***CICLO 1 OCORRÊNCIA 1 do MAU CHEIRO UM***:

- *A primeira coisa que tenho que fazer é : tornar a expressão booleana positiva – logo, o fragmento de código* ***nome != null*** *passa a ser* ***nome == null*** *, e*
- *Sou obrigado a inverter os blocos de código do IF-THEN-ELSE, assim, significa que, eu tenho que inverter aquilo que estava no THEN agora passa a ser o que estava no ELSE, o que vai para o ELSE é o que estava no THEN.*

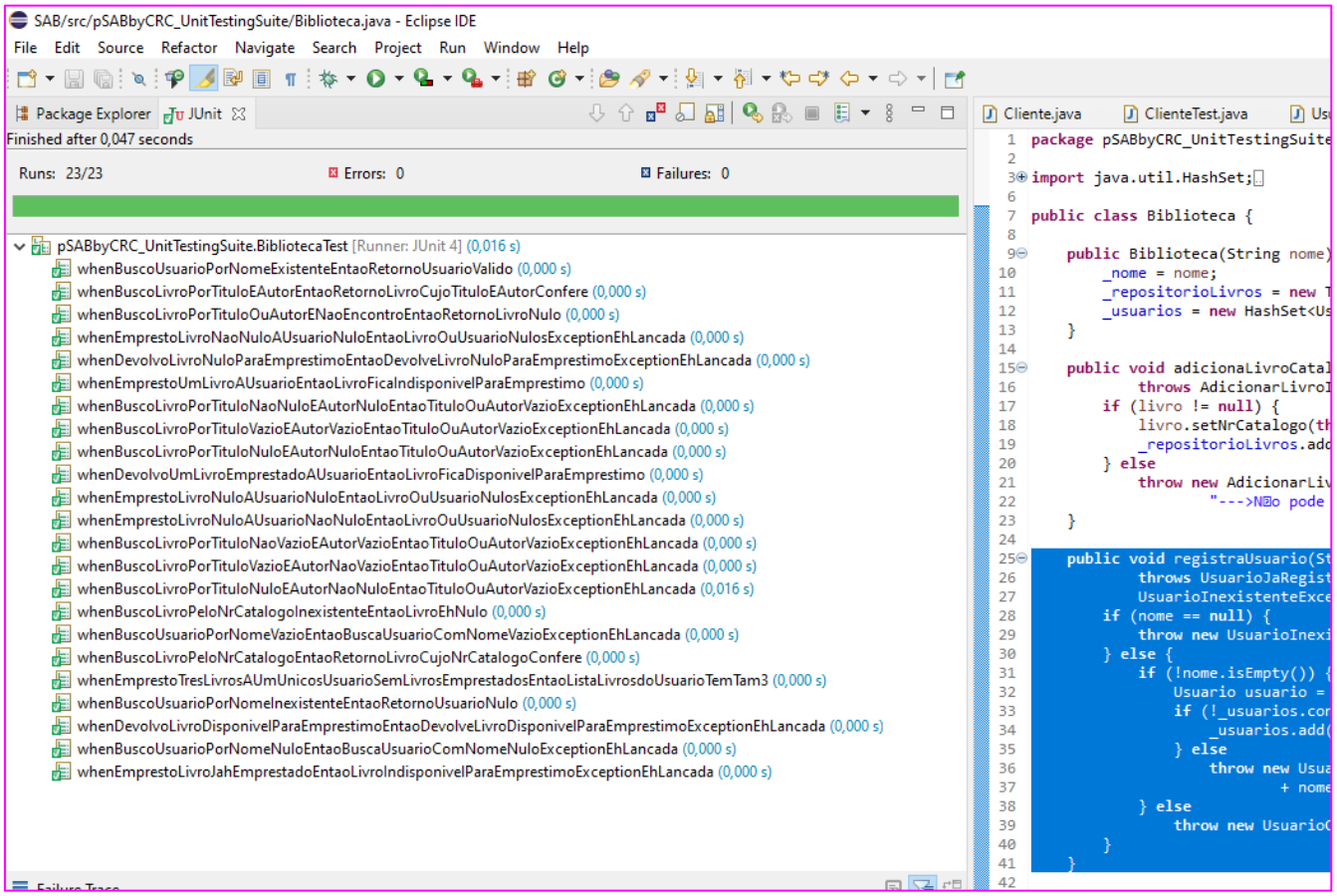
- **Como Ficou o Código Depois da Refatoração:**

```
public void registraUsuario(String nome)
        throws UsuarioJaRegistradoException, UsuarioComNomeVazioException,
                        UsuarioInexistenteException {
    if (nome == null) {
        throw new UsuarioInexistenteException("--->Não pode registrar usuario inexistente!");
    } else {
        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!");
    }
}
```

Aridio Gomes da Silva - aridiosilva@aridiosilva.com - https://www.linkedin.com/in/aridio-silva-74997111/

- **Imagem da Execução Bem-Sucedida (verde) do Código de Produção e Bateria de Testes no Eclipse**

Após a conclusão da realização dos procedimentos para a refatoração do mau cheiro 1 e sua ocorrência 1, e ter sido compilado sem erros e ter rodado a bateria de casos de testes inalterados, e atendendo a exigência de mostrar que os testes passaram sem erro algum, e que, portanto, durante a refatoração não foi de forma inadvertida incluído nenhum erro, respectivamente:



- **Lista de Mau Cheiros Atualizada Após a Refatoração:**

Considerando que a o Mau Cheiro 1 - IFs Com Expressões Negativas ocorre três vezes e que neste CICLO 1 tratamos apenas da OCORRÊNCIA 1, a Lista de Maus Cheiros continua ainda com os três tipos de Maus Cheiros, porém, com o número de ocorrências do Mau Cheiro Um reduzido de um, respectivamente:

#	Tipo do Mau Cheiro Encontrado no Código	Nº de Ocorrências Identificadas
1	*Ifs com Expressões Booleanas Negativas*	*2 (duas)*
2	*Ifs Aninhados*	*3 (três)*
3	*Código Duplicado*	*3 (três)*

Aridio Gomes da Silva - aridiosilva@aridiosilva.com - https://www.linkedin.com/in/aridio-silva-74997111/

V – CONTINUAÇÃO do Ciclo da Refatoração do Código do Método registraUsuario(String)

➢ **Ciclo 2 da Refatoração – Ocorrência 2 do Mau Cheiro 1 - Ifs com Expressões Booleanas Negativas**

- **Antes de Refatorar:**

```
public void registraUsuario(String nome)
        throws UsuarioJaRegistradoException, UsuarioComNomeVazioException,
                        UsuarioInexistenteException {
    if (nome == null) {
        throw new UsuarioInexistenteException("--->Não pode registrar usuario inexistente!");
    } else {
        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!");
    }
}
```

- **Fragmento do Código a Refatorar neste Ciclo 2:**

```
            if (!nome.isEmpty()) {
```

- **Tipos de Técnicas de Refatoração Usadas**

A refatoração para ***eliminação da segunda ocorrência do mau cheiro 1 (If com Expressão Booleana Negativa)*** a ser realizada envolve a adoção das duas técnicas, respectivamente:

- *Trocar as condicionais Negativas para Positivas, uma por vez e*
- *Trocar o bloco de código associado do ELSE para o lugar do bloco de código do IF e vice-versa*

Assim sendo, terei que realizar dois passos neste *ciclo 2*, de modo que não seja alterado o comportamento do código e que possa compilar sem erros, e rodar a bateria de testes de modo que continuem passando. Portanto, procedi o seguinte no processo da refatoração neste ***CICLO 2 OCORRÊNCIA 2 do MAU CHEIRO UM***:

- *A primeira coisa que tenho que fazer é : tornar a expressão booleana positiva – logo, o fragmento de* ***!nome.isEmpty()*** *passa a ser* ***nome.isEmpty()*** *, e*
- *Sou obrigado a inverter os blocos de código do IF-THEN-ELSE, assim, significa que, eu tenho que inverter aquilo que estava no THEN agora passa a ser o que estava no ELSE, o que vai para o ELSE é o que estava no THEN.*

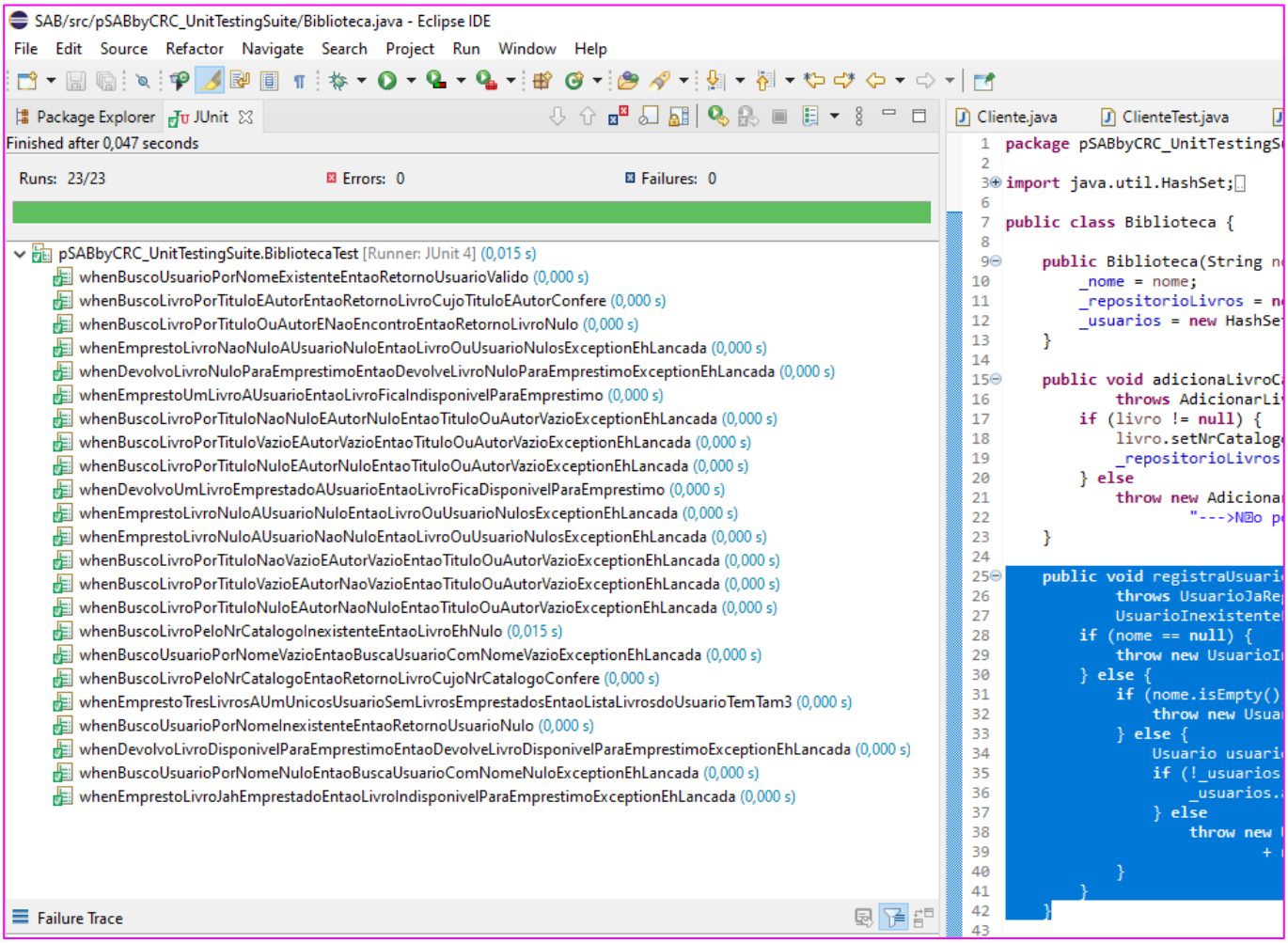
- **Como Ficou o Código Depois da Refatoração:**

```
public void registraUsuario(String nome)
                throws UsuarioJaRegistradoException, UsuarioComNomeVazioException,
                        UsuarioInexistenteException {
    if (nome == null) {
        throw new UsuarioInexistenteException("--->Não pode registrar usuario inexistente!");
    } else {
         if (nome.isEmpty()) {
             throw new UsuarioComNomeVazioException("--->Não pode registrar usuario com nome vazio!");
         } else {
              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!");
         }
    }
}
```

Aridio Gomes da Silva - aridiosilva@aridiosilva.com - https://www.linkedin.com/in/aridio-silva-74997111/

- **Imagem da Execução Bem-Sucedida (verde) do Código de Produção e Bateria de Testes no Eclipse**

Após a conclusão da realização dos procedimentos para a refatoração do mau cheiro 1 e sua ocorrência 2, e ter sido compilado sem erros e ter rodado a bateria de casos de testes inalterados, e atendendo a exigência de mostrar que os testes passaram sem erro algum, e que, portanto, durante a refatoração não foi de forma inadvertida incluído nenhum erro, respectivamente:



- **Lista de Mau Cheiros Atualizada Após a Refatoração:**

Considerando que a o Mau Cheiro 1 - IFs Com Expressões Negativas ocorre três vezes e que neste CICLO 2 tratamos a 2ª CORRÊNCIA, a Lista de Maus Cheiros continua ainda com os um tipo de Maus Cheiros, porém, com o número de ocorrências do Mau Cheiro Um reduzido a apenas uma, respectivamente:

#	Tipo do Mau Cheiro Encontrado no Código	Nº de Ocorrências Identificadas
1	*Ifs com Expressões Booleanas Negativas*	*1 (uma)*
2	*Ifs Aninhados*	*3 (três)*
3	*Código Duplicado*	*3 (três)*

Aridio Gomes da Silva - aridiosilva@aridiosilva.com - https://www.linkedin.com/in/aridio-silva-74997111/

V – CONTINUAÇÃO do Ciclo da Refatoração do Código do Método registraUsuario(String)

➢ **Ciclo 3 da Refatoração – Ocorrência 3 do Mau Cheiro 1 - Ifs com Expressões Booleanas Negativas**

- **Antes de Refatorar:**

```
public void registraUsuario(String nome)
                throws UsuarioJaRegistradoException, UsuarioComNomeVazioException,
                        UsuarioInexistenteException {
    if (nome == null) {
        throw new UsuarioInexistenteException("--->Não pode registrar usuario inexistente!");
    } else {
         if (nome.isEmpty()) {
             throw new UsuarioComNomeVazioException("--->Não pode registrar usuario com nome vazio!");
         } else {
              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!");
         }
    }
}
```

- **Fragmento do Código a Refatorar neste Ciclo 3:**

```
        if (!_usuarios.contains(usuario)) {
```

- **Tipos de Técnicas de Refatoração Usadas**

A refatoração para ***eliminação da terceira ocorrência do mau cheiro 1 (If com Expressão Booleana Negativa)*** a ser realizada envolve a adoção das duas técnicas, respectivamente:

- *Trocar as condicionais Negativas para Positivas, uma por vez e*
- *Trocar o bloco de código associado do ELSE para o lugar do bloco de código do IF e vice-versa*

Assim sendo, terei que realizar dois passos neste *ciclo 3*, de modo que não seja alterado o comportamento do código e que possa compilar sem erros, e rodar a bateria de testes de modo que continuem passando. Portanto, procedi o seguinte no processo da refatoração neste ***CICLO 3 OCORRÊNCIA 3 do MAU CHEIRO UM***:

- *A primeira coisa que tenho que fazer é : tornar a expressão booleana positiva – logo, o fragmento de (!_usuarios.contains(usuario) passa a ser (_usuarios.contains(usuario) , e*
- *Sou obrigado a inverter os blocos de código do IF-THEN-ELSE, assim, significa que, eu tenho que inverter aquilo que estava no THEN agora passa a ser o que estava no ELSE, o que vai para o ELSE é o que estava no THEN.*

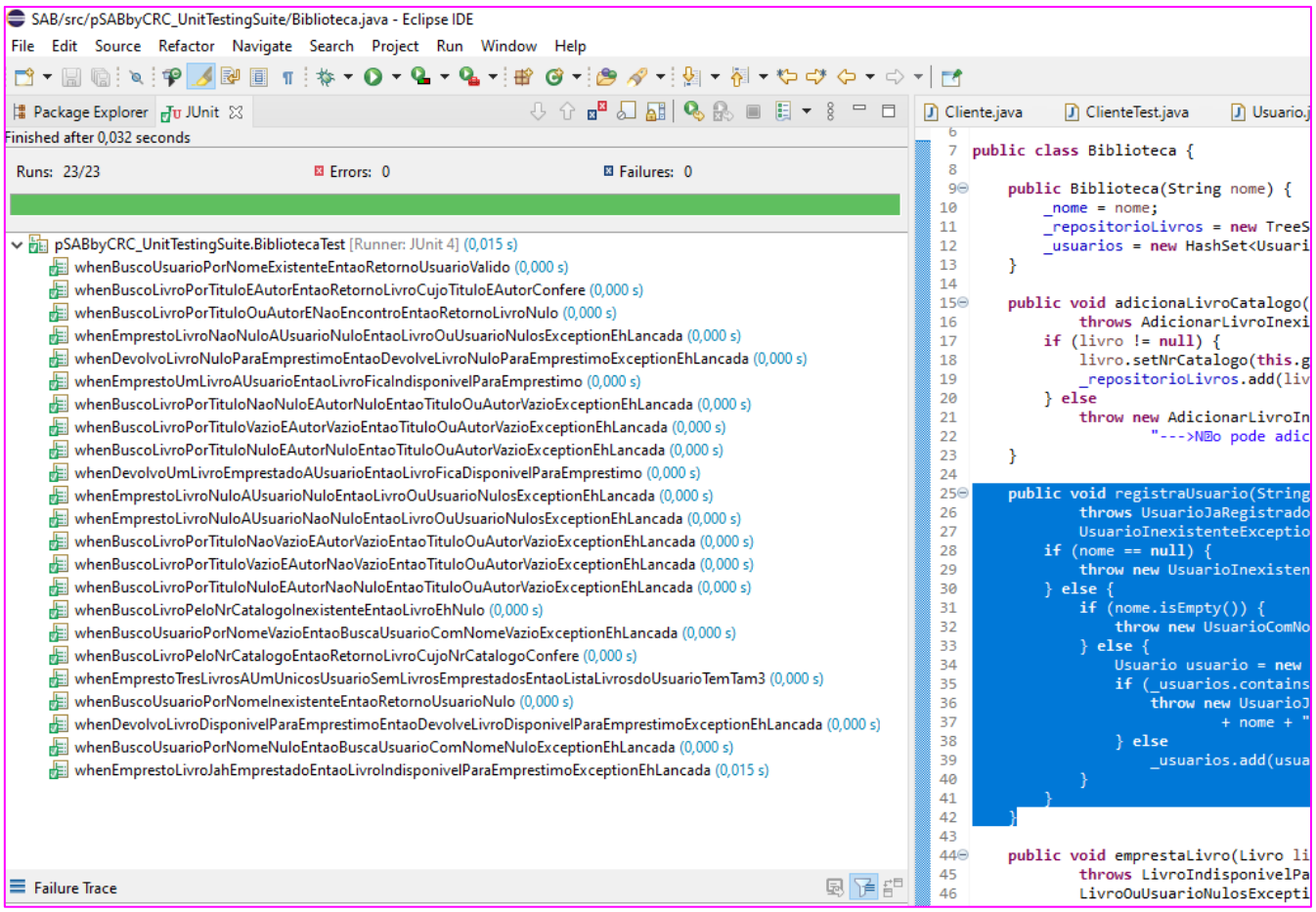
- **Como Ficou o Código Depois da Refatoração:**

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

Aridio Gomes da Silva - aridiosilva@aridiosilva.com - https://www.linkedin.com/in/aridio-silva-74997111/

- **Imagem da Execução Bem-Sucedida (verde) do Código de Produção e Bateria de Testes no Eclipse**

Após a conclusão da realização dos procedimentos para a refatoração do mau cheiro 1 e sua ocorrência 3, e ter sido compilado sem erros e ter rodado a bateria de casos de testes inalterados, e atendendo a exigência de mostrar que os testes passaram sem erro algum, e que, portanto, durante a refatoração não foi de forma inadvertida incluído nenhum erro, respectivamente:



- **Lista de Mau Cheiros Atualizada Após a Refatoração:**

Considerando que a o Mau Cheiro 1 - IFs Com Expressões Negativas ocorre três vezes e que neste CICLO 3 tratamos da OCORRÊNCIA 3 (ÚLTIMA) , a Lista de Maus Cheiros reduz de três para dois tipos de Maus Cheiros remanescentes respectivamente:

#	Tipo do Mau Cheiro Encontrado no Código	Nº de Ocorrências Identificadas
1	*Ifs Aninhados*	*3 (três)*
2	*Código Duplicado*	*3 (três)*

Aridio Gomes da Silva - aridiosilva@aridiosilva.com - https://www.linkedin.com/in/aridio-silva-74997111/

V – CONTINUAÇAO do Ciclo da Refatoração do Código do Método registraUsuario(String)

➢ **Ciclo 4 da Refatoração – Ocorrência 1 do Mau Cheiro 2 : Ifs Aninhados**

- **Antes de Refatorar:**

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

- **Fragmento do Código a Refatorar neste Ciclo 4:**

```
    if (nome == null) {
        throw new UsuarioInexistenteException("--->Não pode registrar usuario inexistente!");
    } else {
        if (nome.isEmpty()) {
```

- **Tipos de Técnicas de Refatoração Usadas**

A refatoração para ***eliminação da primeira ocorrência do mau cheiro 2 (Ifs aninhados)*** a ser realizado envolve a adoção da seguinte técnica, respectivamente:

- *Substituição da condição aninhada com cláusula Guarda*

Assim sendo, terei que realizar um passo neste ciclo 4, de modo que não seja alterado o comportamento do código e que possa compilar sem erros, e rodar a bateria de testes de modo que continuem passando. Portanto, procedi o seguinte no processo da refatoração neste ***CICLO 4 OCORRÊNCIA 1 do MAU CHEIRO DOIS***:

- *A primeira coisa que tenho que fazer é eliminar o código inútil do } else { visto que a comando throws new exception equivale a uma cláusula de guarda, dessa forma equivale a um returno e o fluxo de controle sai do método em questão, porém vai para a classe que trata a respectiva exception;*
- *Precisei corrigir a identação do bloco do código referente ao else eliminado, e assim, tornar o código mais limpo e claro.*

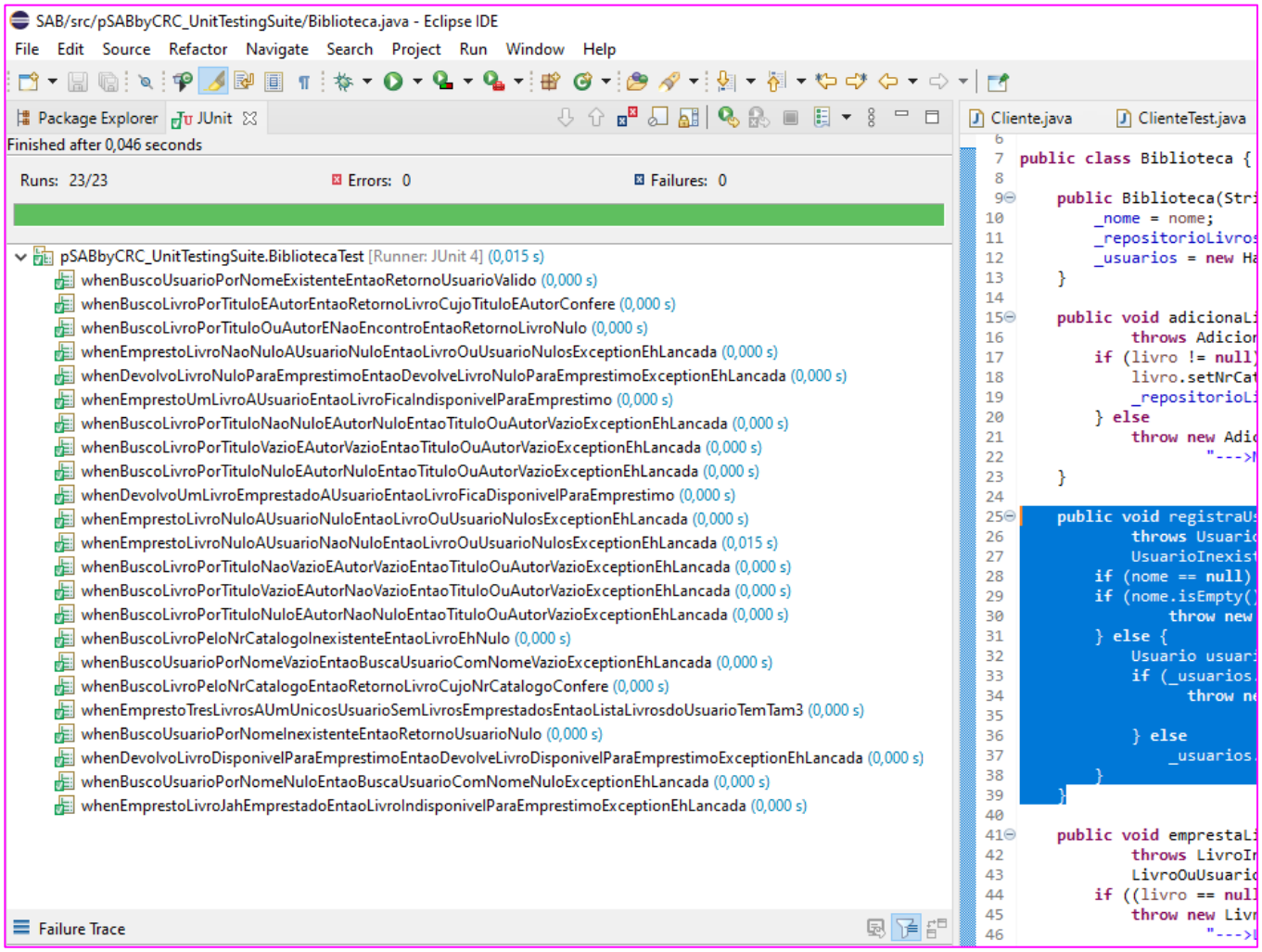
- **Como Ficou o Código Depois da Refatoração:**

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

Aridio Gomes da Silva - aridiosilva@aridiosilva.com - https://www.linkedin.com/in/aridio-silva-74997111/

- **Imagem da Execução Bem-Sucedida (verde) do Código de Produção e Bateria de Testes no Eclipse**

Após a conclusão da realização dos procedimentos para a refatoração do mau cheiro 2 e sua ocorrência 1, e ter sido compilado sem erros e ter rodado a bateria de casos de testes inalterados, e atendendo a exigência de mostrar que os testes passaram sem erro algum, e que, portanto, durante a refatoração não foi de forma inadvertida incluído nenhum erro, respectivamente:



- **Lista de Mau Cheiros Atualizada Após a Refatoração:**

Considerando que a o Mau Cheiro 2 - IFs Aninhados ocorre três vezes e que neste CICLO 4 tratamos apenas da OCORRÊNCIA 1, a Lista de Maus Cheiros continua ainda com os dois tipos de Maus Cheiros, porém, com o número de ocorrências do Mau Cheiro dois reduzido a apenas duas, respectivamente:

#	Tipo do Mau Cheiro Encontrado no Código	Nº de Ocorrências Identificadas
1	*Ifs Aninhados*	*2 (duas)*
2	*Código Duplicado*	*2 (três)*

Importante mencionar que, o código inútil, código morto duplicado, foi automaticamente totalmente sendo eliminado, à medida que, removemos os ELSEs, que ficam desnecessários, após considerar o throws new exception com uma cláusula guarda que possibilita eliminar o anihamento do if seguinte

V – CONTINUAÇAO do Ciclo da Refatoração do Código do Método *registraUsuario(String)*

➢ **Ciclo 5 da Refatoração – Ocorrência 2 do Mau Cheiro 2 : Ifs Aninhados**

- **Antes de Refatorar:**

```
public void registraUsuario(String nome)
                throws UsuarioJaRegistradoException, UsuarioComNomeVazioException,
                       UsuarioInexistenteException {
    if (nome == null)   throw new UsuarioInexistenteException("--->Não pode registrar usuario inexistente!");
    if (nome.isEmpty()) {

          throw new UsuarioComNomeVazioException("--->Não pode registrar usuario com nome vazio!");
    } else {
        Usuario usuario = new Usuario(nome);
        if (_usuarios.contains(usuario)) {
            throw new UsuarioJaRegistradoException("--->Já existe usuário com o nome \""
                            + nome + "\"! Use outro nome!");
        } else
              _usuarios.add(usuario);
    }
}
```

- **Fragmento do Código a Refatorar neste Ciclo 5:**

```
    if (nome.isEmpty()) {
          throw new UsuarioComNomeVazioException("--->Não pode registrar usuario com nome vazio!");
    } else {
```

- **Tipos de Técnicas de Refatoração Usadas**

A refatoração para *eliminação da segunda ocorrência do mau cheiro 2 (Ifs aninhados)* a ser realizado envolve a adoção da seguinte técnica, respectivamente:

- *Substituição da condição aninhada com cláusula Guarda*

Assim sendo, terei que realizar um passo neste ciclo 5, de modo que não seja alterado o comportamento do código e que possa compilar sem erros, e rodar a bateria de testes de modo que continuem passando. Portanto, procedi o seguinte no processo da refatoração neste *CICLO 5 OCORRÊNCIA 2 do MAU CHEIRO DOIS*:

- *A primeira coisa que tenho que fazer é eliminar o código inútil do } else { visto que a comando throws new exception equivale a uma cláusula de guarda, dessa forma equivale a um returno e o fluxo de controle sai do método em questão, porém vai para a classe que trata a respectiva exception;*
- *Precisei corrigir a identação do bloco do código referente ao else eliminado, e assim, tornar o código mais limpo e claro.*

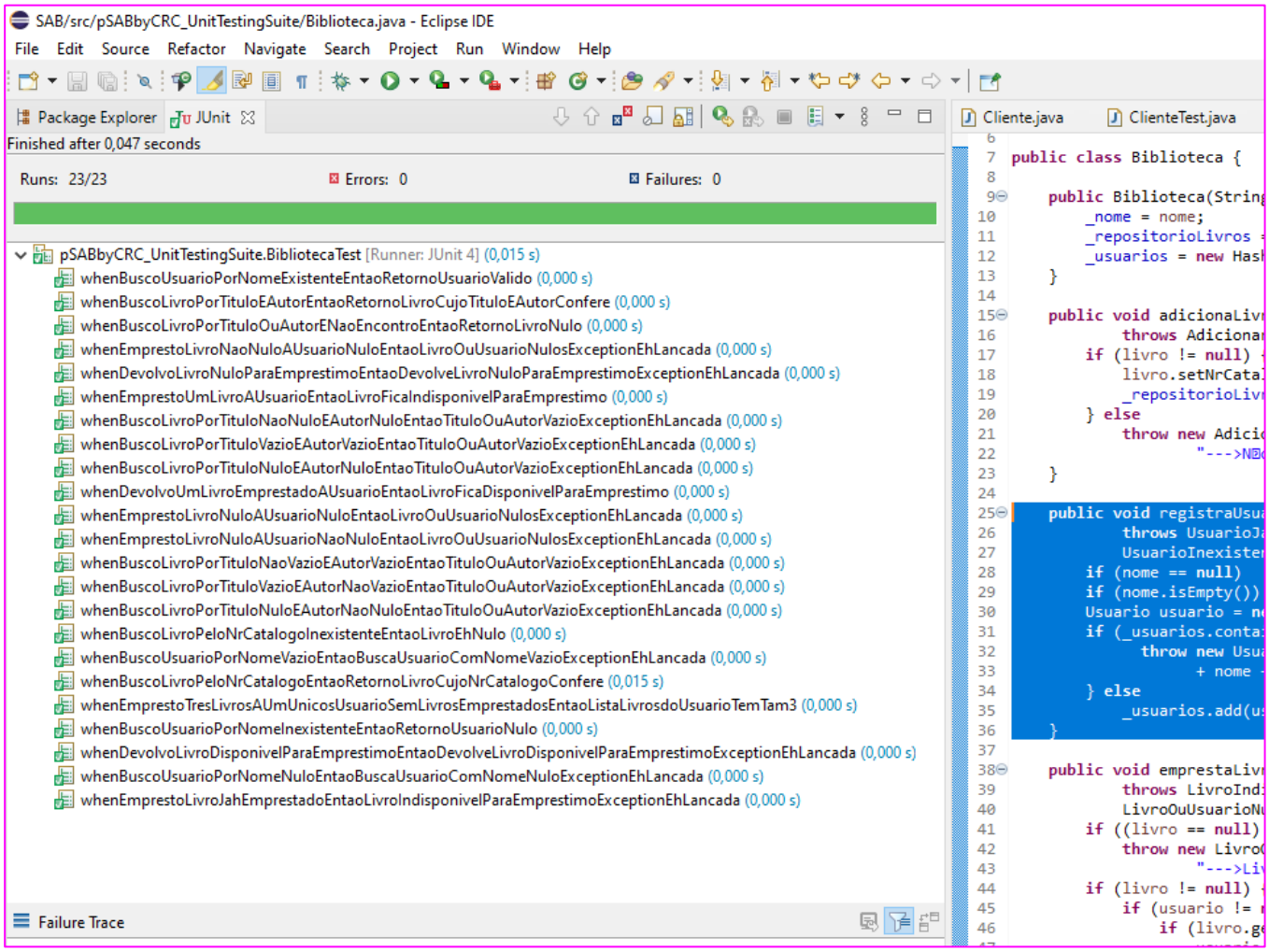
- **Como Ficou o Código Depois da Refatoração:**

```
public void registraUsuario(String nome)
            throws UsuarioJaRegistradoException,UsuarioComNomeVazioException,
                   UsuarioInexistenteException {

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

- **Imagem da Execução Bem-Sucedida (verde) do Código de Produção e Bateria de Testes no Eclipse**

Após a conclusão da realização dos procedimentos para a ***refatoração do mau cheiro 2 e sua ocorrência 2***, e ter sido compilado sem erros e ter rodado a bateria de casos de testes inalterados, e atendendo a exigência de mostrar que os testes passaram sem erro algum, e que, portanto, durante a refatoração não foi de forma inadvertida incluído nenhum erro, respectivamente:



- **Lista de Mau Cheiros Atualizada Após a Refatoração:**

Considerando que a o ***Mau Cheiro 2 - IFs Aninhados*** ocorre ***três vezes*** e que neste ***CICLO 5*** tratamos apenas da ***OCORRÊNCIA 2***, a Lista de Maus Cheiros continua ainda com os dois tipos de Maus Cheiros, porém, com o número de ocorrências do Mau Cheiro dois reduzido a apenas uma, respectivamente:

#	Tipo do Mau Cheiro Encontrado no Código	Nº de Ocorrências Identificadas
1	*Ifs Aninhados*	*1 (uma)*
2	*Código Duplicado*	*1 (uma)*

Importante mencionar que, o código inútil, código morto duplicado, foi automaticamente totalmente sendo eliminado, à medida que, removemos os ELSEs, que ficaram desnecessários, após considerar o comando throws new exception com uma cláusula de guarda - que possibilitou eliminar mais um aninhamento de IF-THEN-ELSE.

Aridio Gomes da Silva - aridiosilva@aridiosilva.com - https://www.linkedin.com/in/aridio-silva-74997111/

V – CONTINUAÇAO do Ciclo da Refatoração do Código do Método registraUsuario(String)

➢ **Ciclo 6 da Refatoração – Ocorrência 3 do Mau Cheiro 2 : Ifs Aninhados**

- **Antes de Refatorar:**

```
public void registraUsuario(String nome)
            throws UsuarioJaRegistradoException,UsuarioComNomeVazioException,
                   UsuarioInexistenteException {

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

- **Fragmento do Código a Refatorar neste Ciclo 6:**

```
if (_usuarios.contains(usuario)) {
    throw new UsuarioJaRegistradoException("--->Já existe usuário com o nome \""
                + nome + "\"! Use outro nome!");
} else
```

- **Tipos de Técnicas de Refatoração Usadas**

A refatoração para *eliminação da terceira ocorrência do mau cheiro 2 (Ifs aninhados)* a ser realizado envolve a adoção da seguinte técnica, respectivamente:

- *Substituição da condição aninhada com cláusula Guarda*

Assim sendo, terei que realizar um passo neste ciclo 5, de modo que não seja alterado o comportamento do código e que possa compilar sem erros, e rodar a bateria de testes de modo que continuem passando. Portanto, procedi o seguinte no processo da refatoração neste *CICLO 5 OCORRÊNCIA 2 do MAU CHEIRO DOIS*:

- *A primeira coisa que tenho que fazer é eliminar o código inútil do } else { visto que a comando throws new exception equivale a uma cláusula de guarda, dessa forma equivale a um returno e o fluxo de controle sai do método em questão, porém vai para a classe que trata a respectiva exception;*
- *Precisei corrigir a identação do bloco do código referente ao else eliminado, e assim, tornar o código mais limpo e claro.*

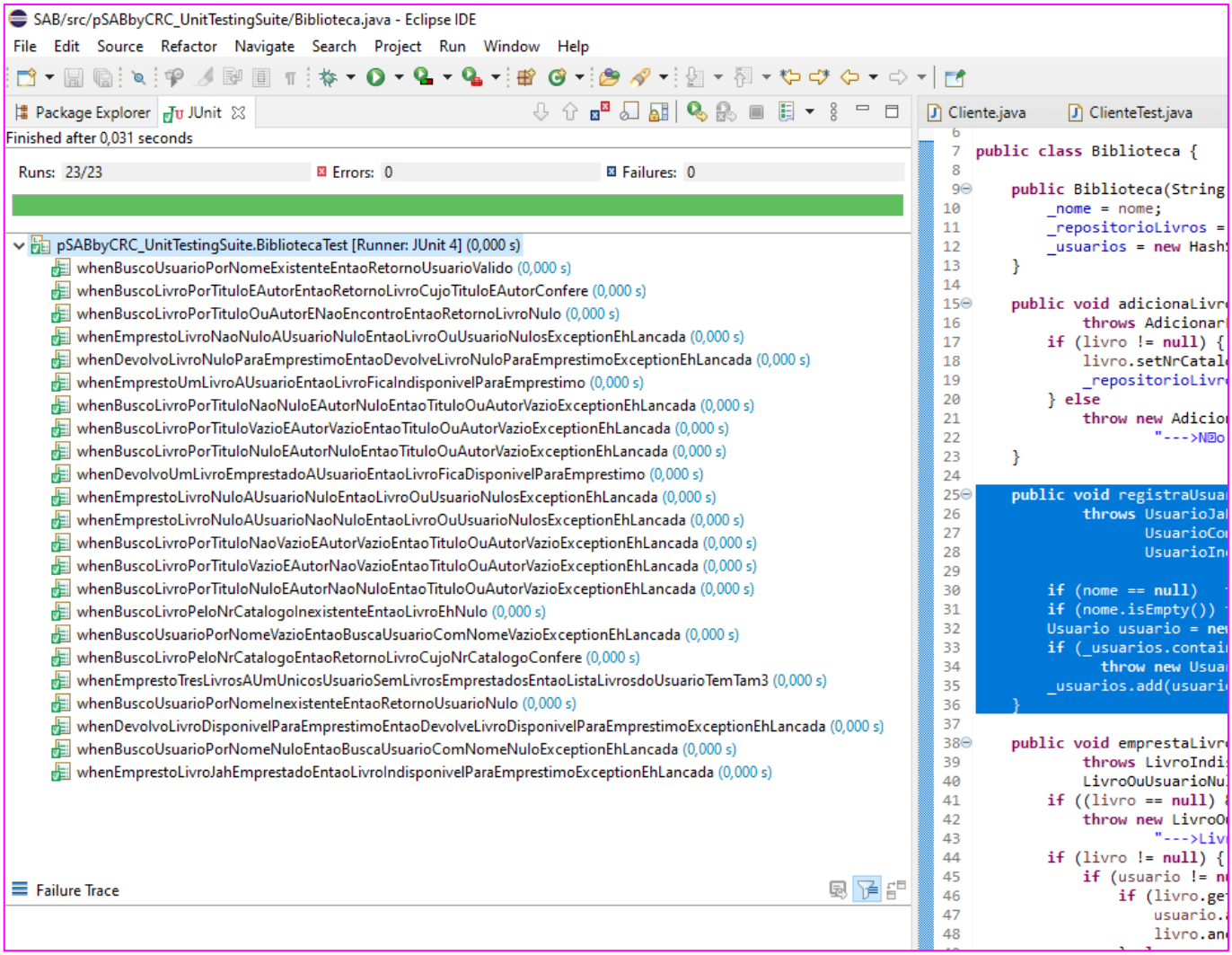
- **Como Ficou o Código Depois da Refatoração:**

```
public void registraUsuario(String nome)
                throws UsuarioJaRegistradoException,
                       UsuarioComNomeVazioException,
                       UsuarioInexistenteException {

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

- **Imagem da Execução Bem-Sucedida (verde) do Código de Produção e Bateria de Testes no Eclipse**

Após a conclusão da realização dos procedimentos para a ***refatoração do mau cheiro 2 e sua ocorrência 2***, e ter sido compilado sem erros e ter rodado a bateria de casos de testes inalterados, e atendendo a exigência de mostrar que os testes passaram sem erro algum, e que, portanto, durante a refatoração não foi de forma inadvertida incluído nenhum erro, respectivamente:



- **Lista de Mau Cheiros Atualizada Após a Refatoração:**

Considerando que a o ***Mau Cheiro 2 - IFs Aninhados*** ocorre ***três vezes*** e que neste ***CICLO 5*** tratamos apenas da ***OCORRÊNCIA 2***, a Lista de Maus Cheiros continua ainda com os dois tipos de Maus Cheiros, porém, com o número de ocorrências do Mau Cheiro dois reduzido a apenas uma, respectivamente:

#	Tipo do Mau Cheiro Encontrado no Código	Nº de Ocorrências Identificadas
LISTA VAZIA		

Importante mencionar que, o código inútil, código morto duplicado, foi automaticamente totalmente sendo eliminado, à medida que, removemos os ELSEs, que ficaram desnecessários, após considerar os comandos throws new exception com cláusulas de guarda - que possibilitou eliminar o aninhamento dos IF-THEN-ELSEs.

V – CONCLUSÃO

Abaixo o comparativo de como o Código era inicialmente, antes da 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 usuário com nome vazio!");
   } else
       throw new UsuarioInexistenteException("--->Não pode registrar usuário inexistente!");
}
```

A agora, após a refatoração:

```
public void registraUsuario(String nome)
               throws UsuarioJaRegistradoException,
                      UsuarioComNomeVazioException,
                      UsuarioInexistenteException {

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

Existe ainda neste Método registroUsuário um bad smell cuja técnica proposta pelo Martin Fowler é o relativo ao uso do Objeto Null no if onde é feita a comparação booloeana com null. Porém, foi requerido que esta tarefa e este documento retratassem, apenas, as técnicas estudadas e apresentadas pelo Professor Clovis Fernandes na SEMANA 2.

Virei fã incondicional da REFATORAÇÃO e estou muito entusiasmado, impressionante como, em ciclos pequenos e integrando com os testes unitários, conseguimos reduzir significativamente a complexidade inerente a um código com vários maus cheiros, mesmo de métodos de pequeno tamanho. No final o código fica limpo, claro, fácil de compreender, e fácil de poder modificar. Realmente uma prática que precisa ser adotada por todos os profissionais da área, integrado com o TDD e com os Testes Unitários, para garantir a qualidade do software, e reduzir o estresse no desenvolvimento de algoritmos e aplicações de qualquer natureza.

Pretendo me aprofundar no estudo de todos os tipos de técnicas de refatoração contidas, e conhecer o maior número possível de bad smells. Já Iniciei a leitura dos Livros: *Refactoring* do *Martin Fowler*; do *Clean Code* do *Robert Martin*; e o *Test-Driven Development* do *Ken Beck*.

Subscrevo-me, cordialmente

Aridio Gomes da Silva
aridiosilva@aridiosilva.com
https://www.linkedin.com/in/aridio-silva-74997111/