



Relatório – Refatoração do SAB

ITA - TDD – Desenvolvimento de Software Guiado por Testes (Coursera)

Aluno: Renato Sampaio Albuquerque

01 – Abaixo iniciamos com o método registraUsuario, que será utilizado para 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!");
}
```

Podemos verificar que os testes propostos pelo próprio modelo SAB passam com sucesso:

The screenshot displays an IDE with two main panels. The left panel shows the 'Package Explorer' with a tree view of test cases under 'pSABByCRC_UnitTestingSuite.BibliotecaTest [Runner: JUnit 4] [0.0.0]'. The test cases listed include 'whenBuscoUsuarioPorNomeExistenteEntaoRetornoUsuarioVal', 'whenBuscoLivroPorTituloEAutorEntaoRetornoLivroCujoTituloE', 'whenBuscoLivroPorTituloOuAutorENaoEncontroEntaoRetornoL', 'whenEmprestoLivroNaoNuloAUuarioNuloEntaoLivroOuUsuari', 'whenDevolveLivroNuloParaEmprestimoEntaoDevolveLivroNulo', 'whenEmprestoUmLivroAUuarioEntaoLivroFicaIndisponivelPar', 'whenBuscoLivroPorTituloNaoNuloEAutorNuloEntaoTituloOuAu', 'whenBuscoLivroPorTituloVazioEAutorVazioEntaoTituloOuAutor', 'whenBuscoLivroPorTituloNuloEAutorNuloEntaoTituloOuAutorV', and 'whenDevolveUmLivroEmprestadoAUuarioEntaoLivroFicaDispo'. The right panel shows the source code for 'Biblioteca.java', specifically the 'registraUsuario(String nome)' method, which matches the code provided in the previous block. The code is highlighted in blue and green, indicating it is the active file.

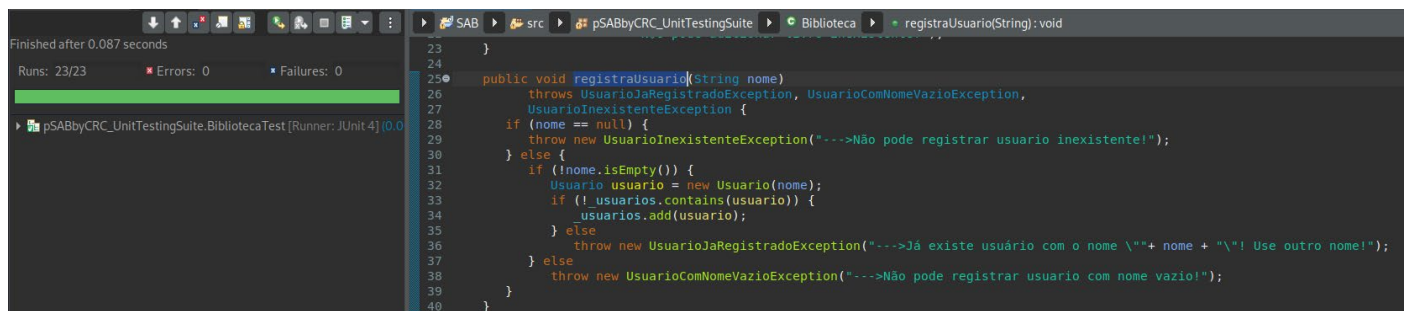
02 – Em sequencia, após inspecionar o código, indicamos a lista de maus cheiros:

Mau Cheiro	Trechos de Código
Condições Negativas	<pre>if (nome != null) if (!nome.isEmpty()) if (!_usuarios.contains(usuario))</pre>
IF's Aninhados	<pre>if (nome != null) { ... if (!nome.isEmpty()) { ... if (!_usuarios.contains(usuario)) {</pre>

03 – Iniciando pelas condições negativas, para ajustar esse problema tornamos as condições positivas e realizamos os ajustes no código:

```
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!");
    }
}
```

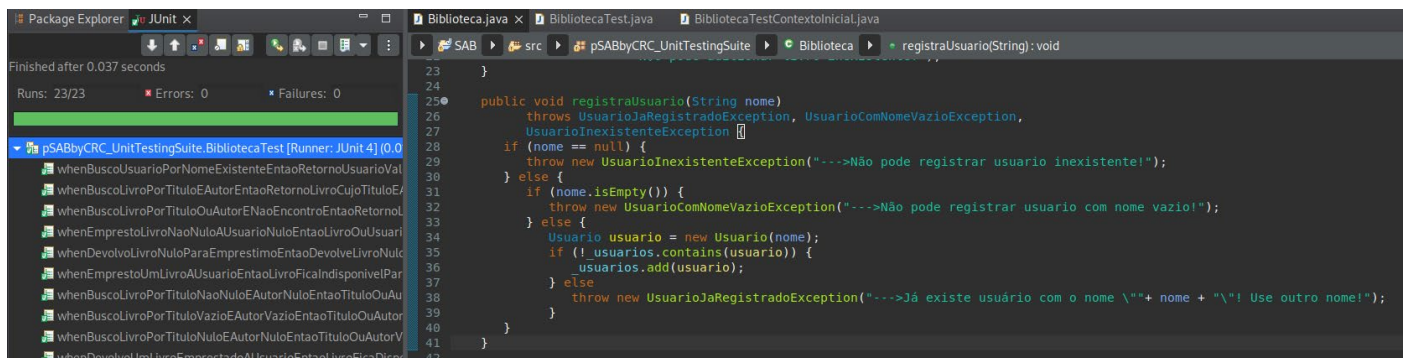
Após a refatoração, verificamos que os testes continuam passando:



The screenshot shows an IDE with a test runner on the left and source code on the right. The test runner indicates 'Finished after 0.087 seconds', 'Runs: 23/23', 'Errors: 0', and 'Failures: 0'. The source code on the right is the refactored version of the `registraUsuario` method, which now uses positive conditions (`nome != null` and `!nome.isEmpty()`) and avoids nested if-statements.

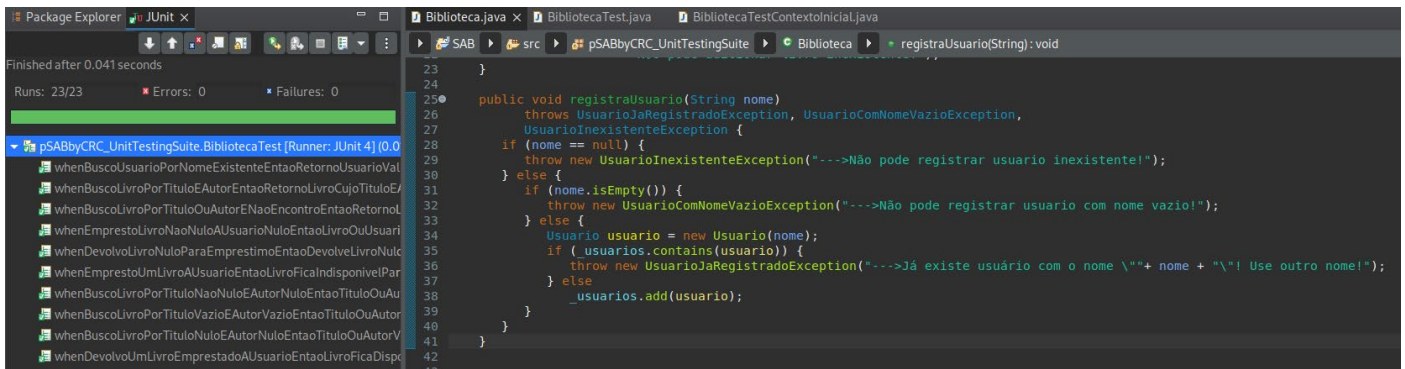
04 – Para um segundo ciclo de refatoração, iremos tornar positivo a segunda condição IF:

```
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!");
        }
    }
}
```



05 – No terceiro ciclo de refatoração iremos tornar positivo a terceira condição IF:

```
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);
        }
    }
}
```



06 – No Com o primeiro mau cheiro eliminado, iremos para o segundo: IF's Aninhados, iremos substituir if/else por cláusulas de guarda.

Aplicando ao IF mais externo:

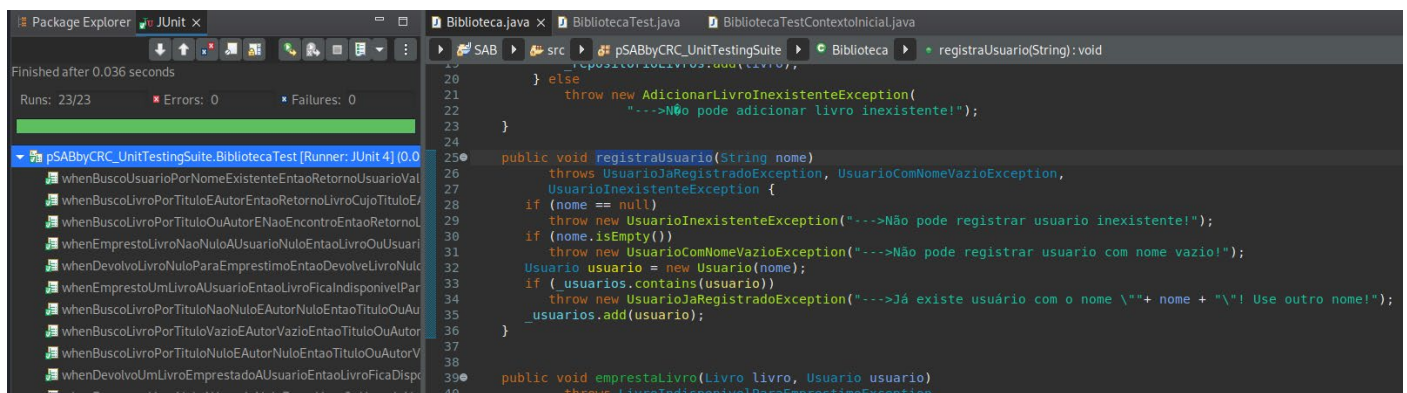
```
public void registraUsuario(String nome)
    throws UsuarioJaRegistradoException, UsuarioComNomeVazioException,
    UsuarioInexistenteException {
    if (nome == null)
        throw new UsuarioInexistenteException("--->Não pode registrar usuario inexistente!");
    if (nome.isEmpty()) {
        throw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

Agora ao segundo IF:

```
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);
}
```

Enfim ao terceiro IF:

```
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 +
"\n"! Use outro nome!");
    _usuarios.add(usuario);
}
```



Após a refatoração, observamos que todos os testes passam. Deixamos o código mais limpo, fácil de ler e mantendo todas as suas funcionalidades.