Faculdade: Universidade Tecnológica Federal do Paraná

Curso: Especialização em Tecnologia Java

Nome: Erik Eduardo Valcezio RA: A2329611

DATA: 30/01/2022

Atividade: Teste de Software

Exercicio2

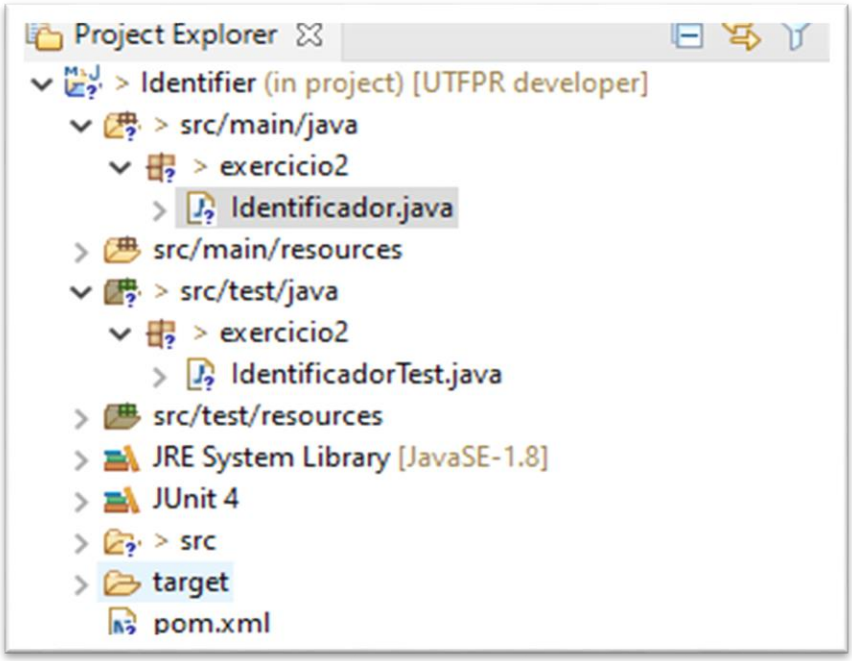
1. Implemente em Java o método para o programa *Identifier.* O método deverá receber o identificador por passagem de valor. Fazer o uso exceções para o tratamento das classes inválidas. Não é necessário fazer a *View* para a entrada dos dados.

2. Implemente no JUnit os casos de teste conforme o particionamento em classes de equivalência.

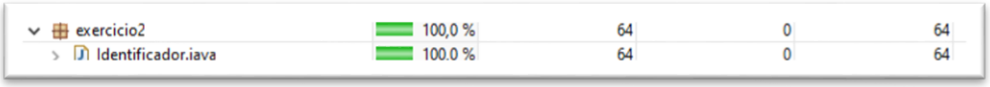
GIT:

https://github.com/erikvalcezio/java-projects-utfpr/tree/develop/Teste-De-Software/Identifier_EquivalenciaExecicio2

Workspace



Cobertura da classe



```java
package exercicio2;

public class Identificador {

    /**
     * O programa deve determinar se um identificador é válido ou não.
     * Um identificador válido deve começar com uma letra e conter apenas letras ou dígitos.
     * Além disso, deve ter no mínimo um caractere e no máximo seis caracteres de comprimento.
     */

    private String entradaDeDados;

    public Identificador (String entradaDeDados) {
        this.entradaDeDados = entradaDeDados;
    }

    /**
     * Getter.
     * @return
     */
    public String getEntradaDeDados() {
        return entradaDeDados;
    }

    /**
     * determinar se um identificador é válido ou não.
     * @param entradaDeDados
     * @return
     * @throws Exception is invalid.
     */
    public boolean validaEntradaDeDados(String entradaDeDados){

        boolean isValidInputData = true;

        if (!verificarTamanhoDoIdentificador(entradaDeDados)) {
            throw new UnsupportedOperationException("Tamanho inválido, deve ser de 1 a 6 caracter(s)!");
        }

        if (!verificarPrimeiroCaracterLetra(entradaDeDados)) {
            throw new UnsupportedOperationException("Identificador inválido, o primeiro caracter deve ser uma letra!");
        }

        if (!verificarCaracterLetraOuNumero(entradaDeDados)) {
            throw new UnsupportedOperationException("Identificador inválido, deve contem apenas letras e números!");
        }
```

```
46
47          return isValidInputData;
48      }
49
50      /**
51       * Mínimo um caractere e no máximo seis caracteres de comprimento.
52       * Tamanho t do identificador.
53       * @return true or false
54       */
55      public boolean verificarTamanhoDoIdentificador(String entradaDeDados) {
56          return entradaDeDados.trim().length() >= 1 &&
57                  entradaDeDados.trim().length() <= 6;
58      }
59
60      /**
61       * Primeiro caracter c é uma letra.
62       * @return
63       */
64      public boolean verificarPrimeiroCaracterLetra(String entradaDeDados) {
65          return Character.isAlphabetic(entradaDeDados.charAt(0));
66      }
67
68      /**
69       * Só contém caracteres válidos
70       * @return
71       */
72      public boolean verificarCaracterLetraOuNumero(String entradaDeDados) {
73          return entradaDeDados.chars().allMatch(Character::isLetterOrDigit);
74      }
75
76  }
77
```

```java
package exercicio2;

import static org.junit.Assert.assertEquals;
import static org.junit.Assert.assertTrue;

import org.junit.Test;
import org.junit.runner.RunWith;
import org.junit.runners.JUnit4;

@RunWith(value = JUnit4.class)
public class IdentificadorTest {

    /**
     * Classe de testes para equivalência exercio2 para Identificador.
     */

    private Identificador identificador;


    /**
     * Tamanho t do identificador, primeiro caracter c é uma letra e só contém caracteres válidos.
     * Cenário 1: (a1,Válido), (1,3,5)
     */
    @Test
    public void testarIdentificadorComValorValido() {
        this.identificador = new Identificador("a1");
            boolean isValid = this.identificador.validaEntradaDeDados(this.identificador.getEntradaDeDados());
            assertTrue(isValid);
    }

    /**
     * Primeiro caracter c é uma letra.
     * Cenário 2: (2B3, Inválido), (4)
     * @throws UnsupportedOperationException is invalid
     */
    @Test
    public void testarIdentificadorComValorDoPrimeiroCaracterInvalido(){
        this.identificador = new Identificador("2B3");
        try{
            this.identificador.validaEntradaDeDados(this.identificador.getEntradaDeDados());
        }catch (UnsupportedOperationException e) {
            final String mensageError = "Identificador inválido, o primeiro caracter deve ser uma letra!";
            assertEquals(mensageError, e.getMessage());
        }
    }

```

```
47      /**
48       * Só contém caracteres válidos.
49       * Cenário 3: (Z-12, Inválido), (6)
50       */
51      @Test
52      public void testarIdentificadorComValorDeCaracteresInvalido(){
53          this.identificador = new Identificador("Z-12");
54          try{
55              this.identificador.validaEntradaDeDados
    (this.identificador.getEntradaDeDados());
56          }catch (UnsupportedOperationException e) {
57              final String mensageError = "Identificador inválido, deve contem
    apenas letras e números!";
58              assertEquals(mensageError, e.getMessage());
59          }
60      }
61
62      /**
63       * (A1b2C3d, Inválido).
64       * Cenário 4: (A1b2C3d, Inválido), (2)
65       */
66      @Test
67      public void testarIdentificadorQuantidadeDeCaracteresInvalido(){
68          this.identificador = new Identificador("A1b2C3d");
69          try{
70              this.identificador.validaEntradaDeDados
    (this.identificador.getEntradaDeDados());
71          }catch (UnsupportedOperationException e) {
72              final String mensageError = "Tamanho inválido, deve ser de 1 a 6
    caracter(s)!";
73              assertEquals(mensageError, e.getMessage());
74          }
75      }
76  }
77
78
79
```