PRUEBAS UNITARIAS CON XUNIT Y C#

JUAN JOSÉ ORREGO ARIAS
JULIÁN ZAPATA ARANGO

TESTING
G – 350

YEISON STIVEN BETANCUR R.

INSTITUCIÓN UNIVERSITARIA PASCUAL BRAVO
FACULTAD DE INGENIERÍA
TECNOLOGÍA EN DESARROLLO DE SOFTWARE
MEDELLÍN
2024

Este taller estuvo muy interesante, recordamos conceptos que habíamos olvidado por el pasar del tiempo, aunque tuvimos varios compliques en el desarrollo del mismo, al momento de ejecutar las pruebas, no nos dejó porque faltaban algunos archivos, y la versión se SDK no era compatible, la actualicé para una reciente y seguía sin funcionar, y la verdad no vimos solución alguna porque no dejaba ejecutar nada.

Pero de igual manera hicimos el proyecto de xUnit y pusimos todos los casos de pruebas, y los verificamos individualmente en otro lugar para corroborar que funcionara y así fue. Al momento de clonar el repositorio también aparecieron algunos errores más que se solucionaron.

Pruebas

En todas las pruebas se crea un objeto de StringOperations para poder llamar los métodos que hay dentro para crear las pruebas de los mismos.

1. Concatenar Strings:

   En esta primera prueba, primero se van a crear dos strings, las cuales van a tener un texto y estos se llevan al método correspondiente para unirlos.

```
[Fact]
//Basicamente acá vamos a crear dos strings en las cuales se les pondrán dos textos
//Y con el Assert, vamos a corroborar de que el resultado del método sea el mismo
public void TestConcatenateStrings()
{
    var stringOperations = new StringOperations();
    string str1 = "Hola";
    string str2 = "Gente";

    string resultado = stringOperations.ConcatenateStrings(str1, str2);

    Assert.Equal("Hola Gente", resultado);
}
```

2. String a la inversa:

   Acá se creará una string con un texto para ponerlo al revés, y luego se compara.

```
[Fact]
//Aquí vamos a crear un string el cual lo que le pongamos se va a poner a la reversa
public void TestReverseString()
{

    var stringOperations = new StringOperations();
    string x = "hola";

    string resultado = stringOperations.ReverseString(x);

    Assert.Equal("aloh", resultado);
}
```

3. Obtener la longitud de un string:

   En esta prueba se va a crear una string la cual tiene un texto, y en este texto se va sacar su longitud, para compararse.

```
[Fact]
//Aquí crearemos una string con un texto, el cual el método nos va a sacar el número de caracteres
public void TestGetStringLength()
{
    var stringOperations = new StringOperations();
    string x = "Hola";

    int tamano = stringOperations.GetStringLength(x);

    Assert.Equal(4, tamano);
}
```

4. Eliminar espacios en blanco:

   Se crea una string con un texto, y en este texto se van a eliminar los espacios en blanco que haya y se queda todo el texto junto.

```
[Fact]
//Crearemos una string para eliminar los espacios que hayan en la misma
public void TestRemoveWhitespace()
{
    var stringOperations = new StringOperations();
    string x = "Liliana Hola";

    string resultado = stringOperations.RemoveWhitespace(x);

    Assert.Equal("LilianaHola", resultado);
}
```

5. Truncar string:

   Se creará una string, y se mandará al método esta string y un número, este número nos sirve para que se acorte la string acorde al número que mandemos.

```
[Fact]
//Crearemos un string y le daremos un valor en especifico para que la
//longitud de la misma se comporte acorde a nuestro interes
public void TestTruncateString()
{
    var stringOperations = new StringOperations();
    string x = "Maria Hola";

    string resultado = stringOperations.TruncateString(x, 5);

    Assert.Equal("Maria", resultado);
}
```

6. String palíndromo:

   Aquí crearemos dos string, una el cual es una palabra palíndroma,y la otra no, para poder ver como se comporta el método con estos dos tipos de palabras.

```
[Fact]
//Aquí se crean dos string, los cuales se corrobora si son Palindromos
//Esto quiere decir, que si se leen de igual manera de atrás para adelate y viceversa
public void TestIsPalindrome()
{
    var stringOperations = new StringOperations();
    string Palindromo = "Oso";
    string NoPalindromo = "Maria";


    Assert.True(stringOperations.IsPalindrome(Palindromo));
    Assert.False(stringOperations.IsPalindrome(NoPalindromo));
}
```

7. Cantidad de caracteres en un string:

   Aquí solo se creará una string, pero al llamar al método, se enviará de igual manera un carácter el cual tiene de propósito corroborar cuantos caracteres de ese tipo hay en esa string.

```
[Fact]
//Se crea un string,y se somete para saber la cantidad de caracteres que hay en el mismo
public void TestCountOccurrences()
{
    var stringOperations = new StringOperations();
    string x = "Hola";

    int resultado = stringOperations.CountOccurrences(x, 'l');

    Assert.Equal(1, resultado);

}
```

8. Pluralizar una string:

   Se creará una String, la cual contendrá una palabra en singular. Al momento de llamar el método, esta palabra en singular se pluralizará.

```
[Fact]
//Pondremos una palabra en singular, y el método la pondrá en plural
public void TestPluralize()
{
    var stringOperations = new StringOperations();
    string Singular = "apple";

    string resultadoSingular = stringOperations.Pluralize(Singular);

    Assert.Equal("apples", resultadoSingular);

}
```

9. Numero a palabras:

   En esta ocasión se va a crear una string y un entero, este entero al llamar el método se va a convertir en palabras.

```
[Fact]
//Aquí crearemos un string y un entero, este entero se va a convertir en palabras
public void TestQuantintyInWords()
{

    var stringOperations = new StringOperations();
    string x = "apples";
    int cantidad = 2;

    string resultado = stringOperations.QuantintyInWords(x, cantidad);

    Assert.Equal("two apples", resultado);
}
```

10. Número romano a número entero:

    Se creará una string el cual contendrá un número en romano, este número cuando se llame el método se convertirá en un entero.

```
[Fact]
//Aquí simplemente se crea un string, el cual tiene un número en Romano, y se convertirá a Entero
public void TestFromRomanToNumber()
{
    var stringOperations = new StringOperations();
    string x = "XIV";

    int resultado = stringOperations.FromRomanToNumber(x);

    Assert.Equal(14, resultado);
}
```