JUnit vs RSpec

Veamos las similitudes y diferencias que existen entre estas 2 herramientas

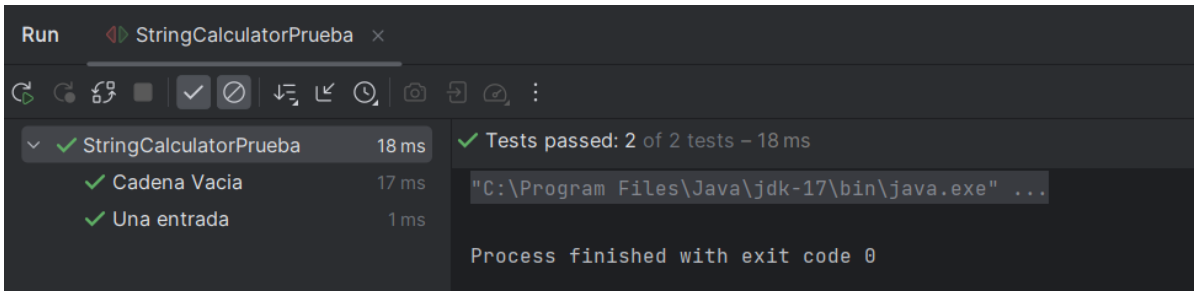
Similitudes

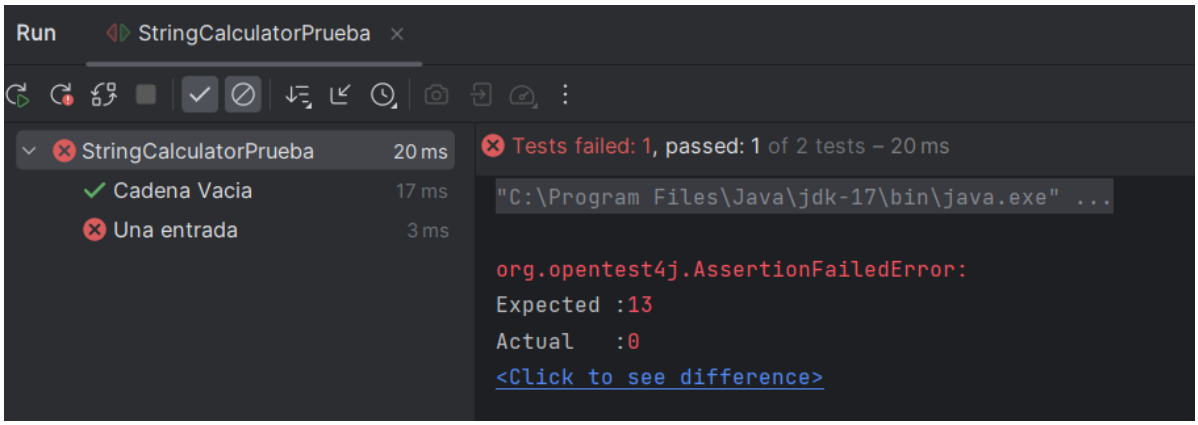
- Ambas estan destinadas para escribir pruebas unitarias, es decir, pruebas para evaluar una clase o un metodo en especifico.
- Generan informes de pruebas
    - RSpec

```
..

Finished in 0.0136 seconds (files took 0.23819 seconds to load)
2 examples, 0 failures
```

```
Failures:

  1) Lista.agregar Se inicializa lista la cantidad debe ser igual a 0
     Failure/Error: expect(@lista.cantidad).to eq(1)

       expected: 1
            got: 0

       (compared using ==)
     # ./spec/Lista_spec.rb:9:in `block (4 levels) in <top (required)>'

Finished in 0.03387 seconds (files took 0.23308 seconds to load)
2 examples, 1 failure

Failed examples:

rspec ./spec/Lista_spec.rb:8 # Lista.agregar Se inicializa lista la cantidad debe ser igual a 0
```

    - JUnit





- En ambas herramientas podemos hacer compraciones mediante los asertions, en este ejemplo ambos verifican que la cantidad de productos de la instancia lista sean iguales a 1.
  - RSpec

```
expect(@lista.cantidad).to eq(1)
```

  - JUnit5

```
assertEquals( expected: 1,l.getCantidad(), message: "Producto agregado");
```

- La estructura es muy similar en el sentido en el que como se estructuran los test
  - Java

```
public class Test_Lista{
  //...
  @Test
  public void prueba0(){
    assertEquals(0,l.getCantidad(),"Lista Vacia");
  }
  @Test
  public void prueba1(){
    l.agregar(p1);
    assertEquals(1,l.getCantidad(),"Se agrega un producto");
  }
  //...
}
```

- RSpec

```
  describe Lista do
    describe ".agregar" do
        # ...
        context "Lista Vacia" do
            expect(@lista.cantidad).to eq(0)
        end
        context "se agrega un producto" do
            @lista.agregar(@producto)
            expect(@lista.cantidad).to eq(1)
        end
        # ...
    end
end
```

Diferencias

- Como ya se pudo haber notado la primera diferencia esta en el lenguaje de programacion, mientras que en JUnit se trabaja con Java en RSpec se trabaja con Ruby.
- Otra diferencia que esta muy relacionada con la anterior es que al ser RSpec escrito en ruby su codigo es mucho mas interpretable por lo cual puede servira mucho mejor como documentacion.
- Los test en JUnit son creadas en clases que se denominan clases de pruebas, mientras que en RSPec es una especie de conjunto de pruebas pero, como vimos en el ejemplo anterior, su sintaxis es la de una clase.