@QuarkusTest  
public class HelloWorldResourceTest {  
  
 @Test  
 public void testHelloEndpoint() {  
 *given*()  
 .when().get("/hello")  
 .then()  
 .statusCode(200)  
 .body(*containsString*("Hello!") );  
 }  
  
}

@QuarkusTest: anotação para dizer que a classe receberá utilizar o framework Quarkus. Quando utilizado essa anotação, a nossa aplicação é iniciada. Utilizado para realizar testes de integração.

@Test: informar que o método é para validação de um Test. Utilizado para realizar testes unitários.

Assertion.assertEquals(valorEsperado, validacao, mensagem);

mvn verify: roda os testes

**Testes unitários:** Validam cenários isolados da sua aplicação.

**Testes de integração:** Validam um cenário mais amplo, com diversas funcionalidades.

**Ordenando testes:**

* Para ordenarmos os testes, devemos adicionar a anotação na classe ***@TestMethodOrder(MethodOrderer.OrderAnnotation.class)***
* Após adicionado esse método na classe, estamos aptos a adicionar a anotação ***@Order(x),*** que servirá para ordenar os métodos, quais irão ser testados primeiro.

***@AfterAll:*** Antes de todo o teste rodar, essa ação será executada;

***@AfterEach:*** Antes de cada teste iniciar, essa ação será executada;

***@BeforeAll:*** Depois de todo o teste rodar, essa ação será executada. Faz com que o método seja static;

***@BeforeEach:*** Depois de cada teste rodar, essa ação será executada;

***@Tag(“database-test”):***

* Essa anotação serve para que criemos tags dos nossos testes, e quando fomos rodar a aplicação, podemos excluir esses tests que possuam tal tag.
  + Adicionar no applitacion.properties o seguinte:
    - Quarkus.test.exclude-tags=database-test
* Funcionalidade apenas do Quarkus >2