



**Certified Tech
Developer**
The Ultimate Degree

Testes Unitários JUnit

Para que são utilizadas as seguintes anotações?

Convidamos você a arrastá-las de acordo com sua definição.

@Test	Cada método precisa ser anotado para que o JUnit o reconheça como um teste e o execute.
@ParameterizedTest	Permite executar o teste com vários argumentos. Pode pegar os parâmetros de diferentes fontes, como um método, valores, um arquivo csv.
@Disable	Desabilita um teste para que ele não seja executado. Um teste com esta anotação será ignorado.
@BeforeEach	Executa um método antes da execução de cada teste.
@AfterEach	Executa um método após a execução de cada teste.
@BeforeAll	Executa um método antes da execução de todos os testes da classe.
@AfterAll	Executa um método após a execução de todos os testes da classe.

Verificação

Mais informações

Entre as anotações que vimos anteriormente também podemos citar:

@Tag: Permite o lançamento de conjuntos de teste com base nas tags que especificamos.

Anotações de **ciclo de vida**: usadas para estabelecer os fixtures. Podem ser de método ou de classe.

Como pudemos ver, JUnit fornece uma grande variedade de asserções que estão localizadas em `org.junit.jupiter.api.Assertions`, por exemplo:

- `assertArrayEquals`
- `assertEquals`
- `assertTrue` and `assertFalse`
- `assertNull` and `assertNotNull`
- `assertSame` and `assertNotSame`
- `assertAll`
- `assertNotEquals`
- `assertIterableEquals`
- `assertThrows`
- `assertTimeout` and `assertTimeoutPreemptively`
- `assertLinesMatch`

```
assertEquals(4, 4);
assertNotEquals(3, 4);
assertTrue(true);
assertFalse(false);
assertNull(null);
assertNotNull("Hello");
assertNotSame(originalObject, otherObject);
```

F.I.R.S.T

Continuando com os testes unitários, vamos conhecer as cinco características que devem ter para serem considerados testes de qualidade. Essas características são conhecidas como princípio F.I.R.S.T.

Vamos ver do que se trata.

