

UNIVERSIDADE FEDERAL DE SERGIPE

DEPARTAMENTO DE COMPUTAÇÃO

TESTE DE SOFTWARE

ISAC KAIK OLIVEIRA SANTOS

Atividade 1

São Cristóvão

2024

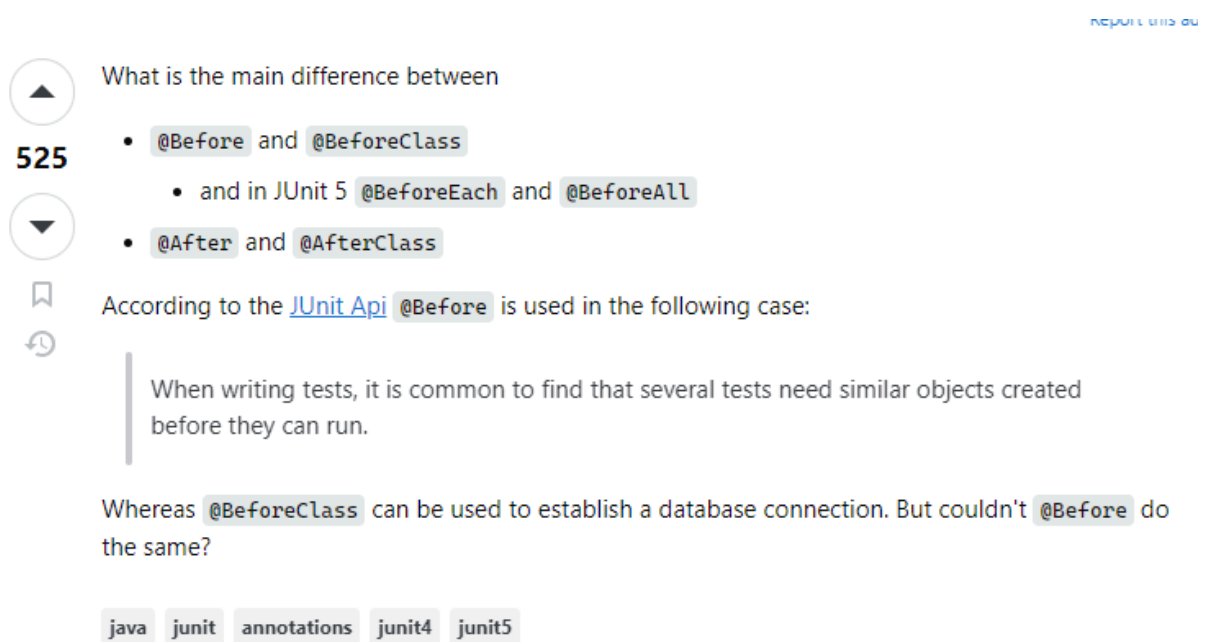
Etapas:

Conforme sugestão dada pelo professor, foi pesquisado no Stack OverFlow problemas envolvendo testes no desenvolvimento de software.

Pesquisando pela string de busca "junit", foi possível encontrar diversos casos, inclusive com mais de 400 votos. Foi escolhido por mim o fórum sobre "Difference between @Before, @BeforeClass, @BeforeEach and @BeforeAll", disponível no endereço:

<https://stackoverflow.com/questions/20295578/difference-between-before-beforeclasses-beforeeach-and-beforeall>

Abaixo segue prints sobre a publicação da pergunta e da resposta marcada com maior pontuação na plataforma.



The screenshot shows a Stack Overflow post. At the top right, there is a link that says "report this as". Below it, on the left, is a circular icon with an upward arrow. To its right is the question text: "What is the main difference between". Below the question text is the number "525". To the right of "525" is a downward arrow icon. Below the downward arrow icon are two bullet points: "`@Before` and `@BeforeClass`" and "`@After` and `@AfterClass`". Below the bullet points is a bookmark icon and a clock icon. To the right of the clock icon is the text: "According to the [JUnit Api](#) `@Before` is used in the following case:". Below this text is a quote box containing the text: "When writing tests, it is common to find that several tests need similar objects created before they can run." Below the quote box is the text: "Whereas `@BeforeClass` can be used to establish a database connection. But couldn't `@Before` do the same?". At the bottom of the screenshot are five tags: "java", "junit", "annotations", "junit4", and "junit5".

report this as

▲

525

▼

- `@Before` and `@BeforeClass`
 - and in JUnit 5 `@BeforeEach` and `@BeforeAll`
- `@After` and `@AfterClass`

🔖

🕒

According to the [JUnit Api](#) `@Before` is used in the following case:

When writing tests, it is common to find that several tests need similar objects created before they can run.

Whereas `@BeforeClass` can be used to establish a database connection. But couldn't `@Before` do the same?

java junit annotations junit4 junit5

7 Answers

Sorted by: Highest score (default)



758

The code marked `@Before` is executed before each test, while `@BeforeClass` runs once before the entire test fixture. If your test class has ten tests, `@Before` code will be executed ten times, but `@BeforeClass` will be executed only once.



In general, you use `@BeforeClass` when multiple tests need to share the same computationally expensive setup code. Establishing a database connection falls into this category. You can move code from `@BeforeClass` into `@Before`, but your test run may take longer. Note that the code marked `@BeforeClass` is run as static initializer, therefore it will run before the class instance of your test fixture is created.

In [JUnit 5](#), the tags `@BeforeEach` and `@BeforeAll` are the equivalents of `@Before` and `@BeforeClass` in JUnit 4. Their names are a bit more indicative of when they run, loosely interpreted: 'before each tests' and 'once before all tests'.

Foi criado um cenário na IDE IntelliJ, usando java 17 com uma pequena aplicação spring Boot com Maven, usando testes com Junit e Mockito para simular as características de cada anotação. Abaixo estão as imagens sobre os códigos de implementação e a saída retornada no console, destacando a ordem de execução dos métodos com as anotações `@BeforeAll` e `@BeforeEach`.

```
public class PessoaServiceTest {

    @InjectMocks 3 usages
    PessoaService service;

    @Mock 7 usages
    PessoaRepository repository;

    Pessoa pessoa; 11 usages

    static List<Pessoa> listaPessoas; 2 usages

    @BeforeAll
    static void setUpAll(){
        System.out.println("Executado BeforeAll antes de todos os métodos de teste.");
        listaPessoas = new ArrayList<>();
    }

    @BeforeEach
    public void setUp(){
        System.out.println("Executado BeforeEach antes de cada método de teste");
        pessoa = Pessoa.builder()
            .nome("Isac")
    }
```

```

public class PessoaServiceTest {
    @BeforeEach
    public void setUp(){
        System.out.println("Executado BeforeEach antes de cada método de teste");
        pessoa = Pessoa.builder()
            .nome("Isac")
            .cpf("12358569852")
            .profissao("Desenvolvedor")
            .idade(24)
            .cidade("Maruim").rua("Rua 1").numero(1)
            .build();
    }

    @Test
    void deveBuscarPessoaPorCpfComSucesso(){
        System.out.println("Executado método de teste 'deveBuscarPessoaPorCpfComSucesso'");
        when(repository.findPessoa(pessoa.getCpf())).thenReturn(Collections.singletonList(pessoa));
        List<Pessoa> pessoas = service.buscaPessoasPorCpf(pessoa.getCpf());

        assertEquals(Collections.singletonList(pessoa), pessoas);
        verify(repository).findPessoa(pessoa.getCpf());
    }

```

```

        assertEquals(Collections.singletonList(pessoa), pessoas);
        verify(repository).findPessoa(pessoa.getCpf());
        verifyNoMoreInteractions(repository);
        listaPessoas.add(pessoa);
    }

    @Test
    void naoDeveChamarRepositorySeCpfNulo(){
        System.out.println("Executado método de teste 'naoDeveChamarRepositorySeCpfNulo'");
        final BusinessException e = assertThrows(BusinessException.class, () -> {
            service.buscaPessoasPorCpf(null);
        });

        assertNotNull(e);
        assertEquals(e.getMessage(), "Erro ao buscar pessoas por cpf = null");
        assertNotNull(e.getCause());
        assertEquals(e.getCause().getMessage(), "Cpf é obrigatório");
        verifyNoInteractions(repository);
    }

```

```

    @Test
    void deveRetornarExceptionSeRepositoryFalhar(){
        System.out.println("Executado método de teste 'deveRetornarExceptionSeRepositoryFalhar'");
        when(repository.findPessoa(pessoa.getCpf())).thenThrow(new RuntimeException("Falha ao buscar pessoas por cpf!"));

        final BusinessException e = assertThrows(BusinessException.class, () -> {
            service.buscaPessoasPorCpf(pessoa.getCpf());
        });

        assertEquals(e.getMessage(), format("Erro ao buscar pessoas por cpf = %s", pessoa.getCpf()));
        assertEquals(e.getCause().getClass(), RuntimeException.class);
        assertEquals(e.getCause().getMessage(), "Falha ao buscar pessoas por cpf!");
        verify(repository).findPessoa(pessoa.getCpf());
        verifyNoMoreInteractions(repository);
    }

```

```
✓ PessoaServiceTest (0.1sec 531ms) ✓ Tests passed: 3 of 3 tests - 1sec 531ms
✓ naoDeveChamarRe 1 sec 455ms
✓ deveBuscarPessoaPorCp 65ms
✓ deveRetornarExceptionSe 11ms

"C:\Program Files\Java\jdk-21\bin\java.exe" ...
Executado BeforeAll antes de todos os métodos de teste.
WARNING: A Java agent has been loaded dynamically (C:\Users\Isac - Agape\.m2\repository\net\bytebuddy\byte-buddy-agent\1.14.18\byte-buddy-agent-1.14.18.jar)
WARNING: If a serviceability tool is in use, please run with -XX:+EnableDynamicAgentLoading to hide this warning
WARNING: If a serviceability tool is not in use, please run with -Djdk.instrument.traceUsage for more information
WARNING: Dynamic loading of agents will be disallowed by default in a future release
Java HotSpot(TM) 64-Bit Server VM warning: Sharing is only supported for boot loader classes because bootstrap classpath has been appended
Executado BeforeEach antes de cada método de teste
Executado método de teste 'naoDeveChamarRepositorySeCpfNulo'
Executado BeforeEach antes de cada método de teste
Executado método de teste 'deveBuscarPessoaPorCpfComSucesso'
Executado BeforeEach antes de cada método de teste
Executado método de teste 'deveRetornarExceptionSeRepositoryFalhar'

Process finished with exit code 0
```

Acima, podemos observar que o método com o `@BeforeAll` foi chamado apenas uma vez, antes de todos os métodos, e que o método com o `@BeforeEach` foi chamado uma vez antes de cada método de teste.

Link do repositório no GitHub:

https://github.com/isackaik/Teste_Software_2024_santos_isac