

Rafael Takeguma Goto

Atividade 1

Testes Unitários e o Stack Overflow

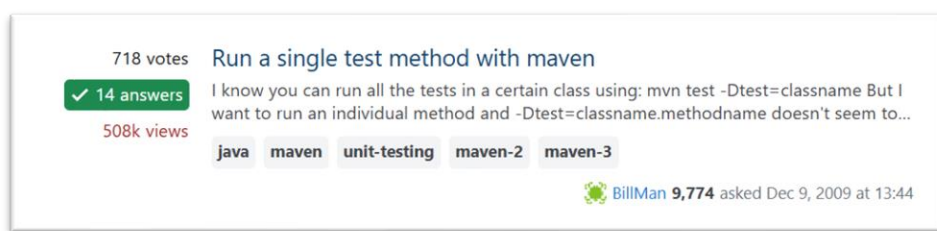
São Cristóvão
08/2024

Busca no Stack Overflow

Para o propósito desta atividade, foi necessário buscar e selecionar uma pergunta no Stack Overflow, relacionada a testes de unidade. Dessa maneira, foi realizada uma busca na barra de pesquisa do website com a seguinte string de busca: [unit-testing].

A fim de atender o critério pré-estabelecido, de 400 votos mínimos, para a seleção da pergunta, foi utilizado o filtro “Highest score” para visualizar as perguntas em ordem decrescente com relação ao número de votos. Desse modo, foi possível encontrar uma pergunta relacionada a execução de testes de unidade por meio de linha de comando usando Maven. A pergunta também possui uma resposta certa, ou seja, atende aos dois critérios pré-estabelecidos.

Figura 01: Pergunta encontrada com a string de busca [unit-testing]



Fonte: [Stack Overflow](#)

Pergunta Escolhida

A pergunta escolhida para a realização desta atividade envolve execução de testes com maven, ferramenta de gerenciamento de projetos Java. Realizada por BillMan, a pergunta possui 718 votos e uma resposta correta.

Esta pergunta foi selecionada por atender aos critérios pré-estabelecidos e por tratar diretamente com testes unitários de métodos de testes definidos em classes, de modo que é possível pôr em prática conceitos vistos em aula, no que tange testes unitários.

Figura 02: Pergunta escolhida no Stack Overflow



Fonte: [Stack Overflow](#)

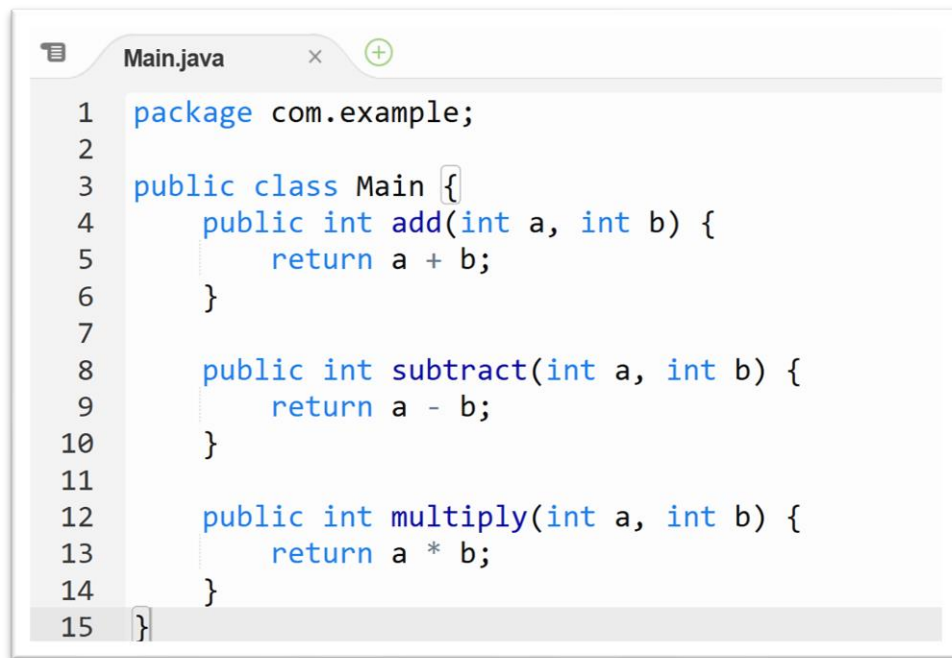
Cadastrada em 2009, essa pergunta busca o procedimento que deve ser realizado para executar um método de teste em específico com o comando mvn, sendo o comando para realizar todos os métodos de uma classe previamente conhecido: `mvn test -Dtest=classname`.

Reprodução do Problema

Para reproduzir e descrever o problema em questão, foi utilizado o AWS Cloud9, IDE em nuvem da AWS. De modo a ilustrar a situação em que existem vários métodos de teste, foram criadas duas classes com extensão java: Main e MyTest.

O arquivo Main.java é uma classe que define métodos básicos para as operações aritméticas de adição, subtração e multiplicação. Esta é a classe que foi testada por meio de testes unitários, definidos em MyTest.java.

Figura 03: Arquivo Main.java



```
1 package com.example;
2
3 public class Main {
4     public int add(int a, int b) {
5         return a + b;
6     }
7
8     public int subtract(int a, int b) {
9         return a - b;
10    }
11
12    public int multiply(int a, int b) {
13        return a * b;
14    }
15 }
```

Fonte: Autor

Para testar os métodos definidos na classe Main, cada um deles possui um teste unitário respectivo, implementado no arquivo MyTest.java. Os métodos de teste definidos na classe MyTest tem a função de verificar se as unidades da classe Main funcionam conforme o esperado.

Todos os métodos definidos em MyTest possuem a anotação @Test para denotar que são métodos de teste. Ademais, todos os métodos de teste possuem o comando assertEquals, que verifica se o resultado obtido é coerente com o resultado esperado.

Figura 04: Arquivo MyTest.java

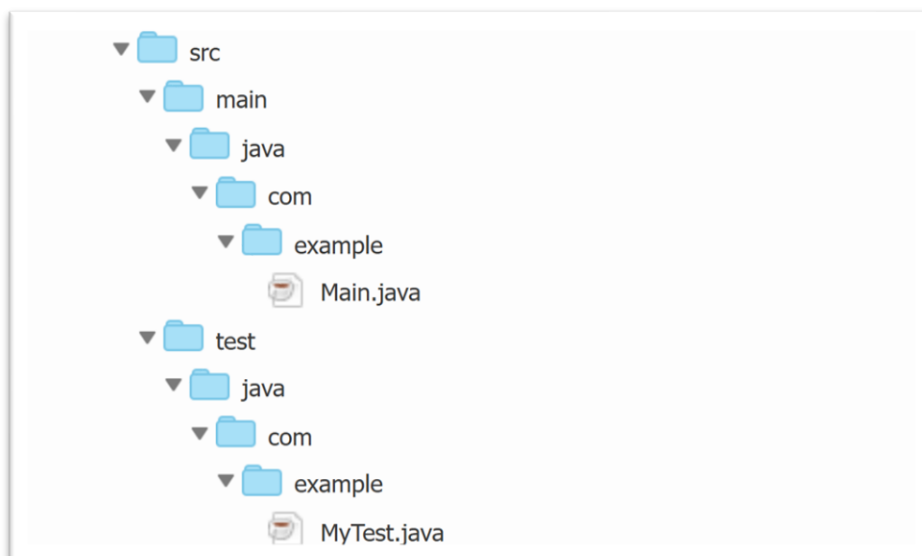


```
1 package com.example;
2
3 import static org.junit.Assert.assertEquals;
4
5 import org.junit.Test;
6
7 public class MyTest {
8     @Test
9     public void testAdd() {           // Método de teste para add()
10         Main main = new Main();
11         int result = main.add(2, 3);
12         assertEquals(5, result);
13     }
14
15     @Test
16     public void testSubtract() {      // Método de teste para subtract()
17         Main main = new Main();
18         int result = main.subtract(5, 3);
19         assertEquals(2, result);
20     }
21
22     @Test
23     public void testMultiply() {      // Método de teste para multiply()
24         Main main = new Main();
25         int result = main.multiply(2, 3);
26         assertEquals(6, result);
27     }
28 }
```

Fonte: Autor

Com o intuito de seguir o layout de diretórios padrão recomendado no [site oficial do Apache Maven](#), a estrutura de diretórios e arquivos do projeto está organizada conforme apresentado na figura 05.

Figura 05: Estrutura do projeto



Fonte: Autor

Para configurar o JUnit no projeto Maven, foi adicionada a dependência de JUnit no arquivo pom.xml. A figura 06 mostra essa adição no pom.xml do projeto.

Figura 06: Dependência do JUnit em pom.xml



Fonte: Autor

Para reproduzir o comando de executar todos os testes de uma classe, foi necessário fazer a instalação do Maven, por meio de `sudo apt install maven`.

Figura 07: Instalação do maven



Fonte: Autor

Utilizou-se o comando `mvn -version` para se certificar que o maven foi de fato instalado com sucesso. A versão utilizada nesta atividade é a Apache Maven 3.6.3.

Figura 08: Versão do maven

A terminal window titled 'bash - "ip-172-31"' with a green plus icon. The prompt is 'voclabs:~/environment \$'. The command 'mvn --version' has been executed, displaying the following information: 'Apache Maven 3.6.3', 'Maven home: /usr/share/maven', 'Java version: 11.0.23, vendor: Ubuntu, runtime: /usr/lib/jvm/java-11-openjdk-amd64', 'Default locale: en, platform encoding: UTF-8', and 'OS name: "linux", version: "6.5.0-1023-aws", arch: "amd64", family: "unix"'. The prompt returns to 'voclabs:~/environment \$' with a cursor.

```
bash - "ip-172-31"
voclabs:~/environment $ mvn --version
Apache Maven 3.6.3
Maven home: /usr/share/maven
Java version: 11.0.23, vendor: Ubuntu, runtime: /usr/lib/jvm/java-11-openjdk-amd64
Default locale: en, platform encoding: UTF-8
OS name: "linux", version: "6.5.0-1023-aws", arch: "amd64", family: "unix"
voclabs:~/environment $
```

Fonte: Autor

Uma vez concluída a configuração do ambiente de desenvolvimento, foi possível reproduzir o comando que o usuário informou ter conhecimento, na pergunta cadastrada no Stack Overflow.

Segundo o usuário, o comando `mvn test -Dtest=nome_da_classe` executa todos os métodos de teste definidos em uma classe. A figura 09 demonstra a execução deste comando para a classe `MyTest`, no AWS Cloud9. No total foram realizados três testes, com zero falhas e zero erros.

Figura 09: Resultados obtidos por meio do comando `mvn test -Dtest=MyTest`A terminal window showing the output of the command 'mvn test -Dtest=MyTest'. The output is color-coded and includes the following information: '[INFO] -----', '[INFO] T E S T S', '[INFO] -----', '[INFO] Running com.example.MyTest', '[INFO] Tests run: 3, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.163 s - in com.example.MyTest', '[INFO] Results:', '[INFO] Tests run: 3, Failures: 0, Errors: 0, Skipped: 0', '[INFO] -----', '[INFO] BUILD SUCCESS', '[INFO] -----', '[INFO] Total time: 3.353 s', '[INFO] Finished at: 2024-08-06T15:59:39Z', and '[INFO] -----'.

```
[INFO] -----
[INFO] T E S T S
[INFO] -----
[INFO] Running com.example.MyTest
[INFO] Tests run: 3, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.163 s - in com.example.MyTest
[INFO] Results:
[INFO] Tests run: 3, Failures: 0, Errors: 0, Skipped: 0
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 3.353 s
[INFO] Finished at: 2024-08-06T15:59:39Z
[INFO] -----
```

Fonte: Autor

Solução

O problema enfrentado pelo usuário, no entanto, é que ele gostaria de fazer um procedimento na linha de comando, similar ao de executar todos os testes, porém para apenas um teste em específico.

A resposta certa escolhida pelo Stack Overflow, traz uma solução para o problema conforme requisitado pelo usuário que realizou a pergunta. Com 959 votos, a resposta em questão, explica como utilizar o comando mvn para executar um teste específico de uma determinada classe, por meio da linha de comando.

Figura 10: Resposta certa escolhida pelo Stack Overflow



Fonte: [Stack Overflow](#)

De acordo com essa resposta, é possível adicionar o nome do teste que se deseja executar com #nome_do_teste no final do comando que foi utilizado até então, mvn test -Dtest=nome_da_classe. Desse modo, apenas o teste especificado será executado.

Para aplicar a resposta escolhida no ambiente configurado, foi escolhido o método testMultiply() da classe MyTest. A figura 11 mostra os resultados obtidos. Pode-se observar que, diferentemente da execução anterior, que englobava todos

os testes, a execução de apenas um dos métodos de teste deixou de gerar três execuções bem-sucedidas, e passou a gerar apenas uma.

Figura 11: Resultados obtidos por meio do comando `mvn test -Dtest=MyTest#testMultiply`

```
[INFO] -----  
[INFO] T E S T S  
[INFO] -----  
[INFO] Running com.example.MyTest  
[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.157 s - in com.example.MyTest  
[INFO]  
[INFO] Results:  
[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0  
[INFO]  
[INFO] -----  
[INFO] BUILD SUCCESS  
[INFO] -----  
[INFO] Total time: 3.935 s  
[INFO] Finished at: 2024-08-06T16:52:23Z  
[INFO]
```


Fonte: Autor

Outras respostas

Esta seção é destinada a discussão das demais respostas que não foram classificadas como corretas pelo Stack Overflow. Além da resposta que foi reproduzida, a pergunta em questão recebeu outras 13 respostas.

Esta segunda resposta traz uma solução idêntica a primeira, todavia foi respondida no ano de 2020, enquanto a correta foi cadastrada em 2011. Além de resolver o problema em questão, o usuário que escreveu a resposta, também trouxe soluções para casos similares.

Figura 12: Resposta cadastrada por Amit, em julho de 2020



The screenshot shows a Stack Overflow post with an upvote icon and a score of 29. The title is "Run a single test method from a test class." Below the title is a code block containing the command: `mvn test -Dtest=Test1#methodName`. Underneath is a section titled "Other related use-cases" with a list of five bullet points, each followed by a code snippet. At the bottom right, it says "answered Jul 20, 2020 at 16:12" and shows the user's profile picture, name "Amit", and statistics: 34.4k, 92, 230, and 301.

▲ **Run a single test method from a test class.**

29 `mvn test -Dtest=Test1#methodName`

▼

Other related use-cases

- `mvn test` // Run all the unit test classes
- `mvn test -Dtest=Test1` // Run a single test class
- `mvn test -Dtest=Test1,Test2` // Run multiple test classes
- `mvn test -Dtest=Test1#testFoo*` // Run all test methods that match pattern 'testFoo*' from a test class.
- `mvn test -Dtest=Test1#testFoo*+testBar*` // Run all test methods match pattern 'testFoo*' and 'testBar*' from a test class.

Share Edit Follow

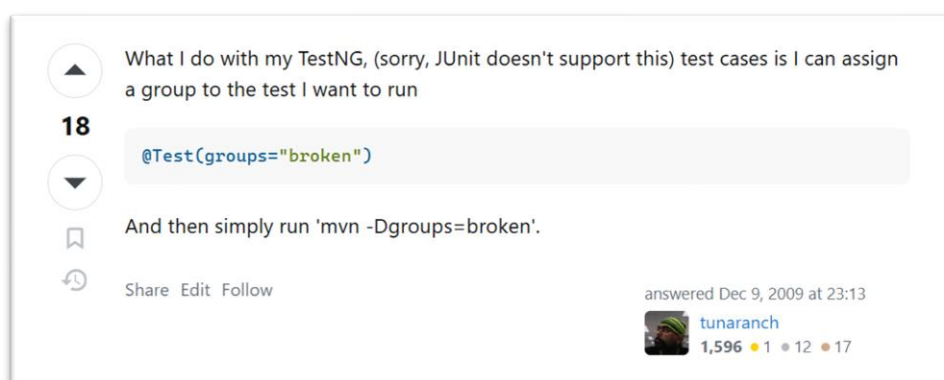
answered Jul 20, 2020 at 16:12

 Amit
34.4k ● 92 ● 230 ● 301

Fonte: [Stack Overflow](#)

A próxima resposta traz uma solução com um framework diferente do JUnit, o TestNG. Esta resposta foi cadastrada no mesmo dia da pergunta, contudo não foi classificada como correta. Um possível motivo que impediu que essa resposta fosse a escolhida, é o fato de que foi necessário criar um grupo que agrupasse os testes, de modo que não seria a solução mais simples e direta.

Figura 13: Resposta cadastrada por tunaranch, em dezembro de 2009



The screenshot shows a Stack Overflow post with an upvote icon and a score of 18. The text describes a solution using TestNG, mentioning that JUnit doesn't support this. It includes a code block with `@Test(groups="broken")` and another line of text about running 'mvn -Dgroups=broken'. At the bottom right, it says "answered Dec 9, 2009 at 23:13" and shows the user's profile picture, name "tunaranch", and statistics: 1,596, 1, 12, and 17.

▲ What I do with my TestNG, (sorry, JUnit doesn't support this) test cases is I can assign a group to the test I want to run

18


▼

`@Test(groups="broken")`

And then simply run 'mvn -Dgroups=broken'.

Share Edit Follow

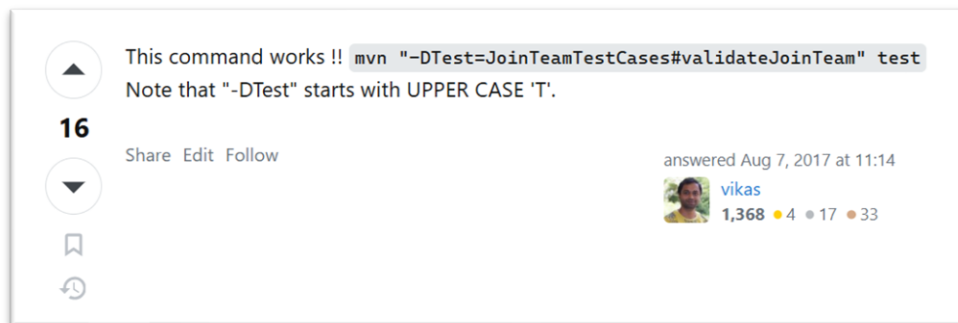
answered Dec 9, 2009 at 23:13

 tunaranch
1,596 ● 1 ● 12 ● 17

Fonte: [Stack Overflow](#)

A próxima solução é similar à escolhida, com uma mudança sutil na sintaxe do comando. De acordo com essa resposta, o comando para solucionar o exemplo da classe MyTest seria `mvn "-DTest=MyTest#testMultiply test"`. A possível causa que torna essa resposta pior do que a escolhida, é porque a sintaxe utilizada não é a forma padrão de executar testes com o Maven.

Figura 14: Resposta cadastrada por vikas, em agosto de 2017



Fonte: [Stack Overflow](#)

Repositório no GitHub

O repositório no GitHub criado para esta atividade pode ser acessado por meio do seguinte link: https://github.com/rafaseto/Teste_Software_2024_Goto_Rafael.git

Vídeo Tutorial

O vídeo tutorial elaborado para esta atividade pode ser acessado por meio do seguinte link: <https://drive.google.com/file/d/1KBgNF1lpdqOupgg15Cz0JJQ2o-oZTpPn/view?usp=sharing>