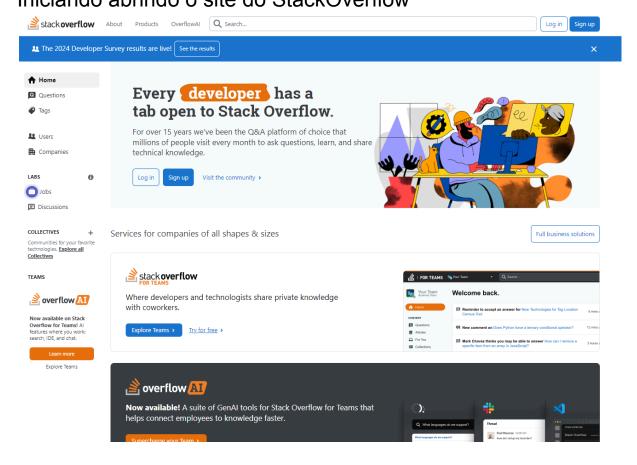
Teste de Software

IAN DA SILVA SANTOS CONCEIÇÃO

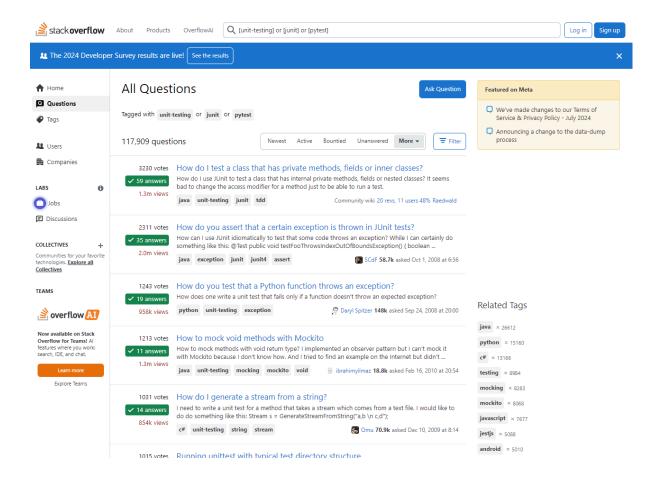
Parte 1) Iniciando abrindo o site do StackOverflow



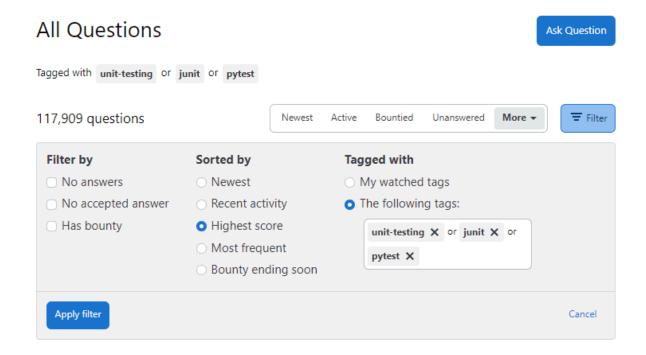
vamos na barra de pesquisa e adicionamos nosso filtro para obter mais precisão.

O filtro utilizado foi: "[unit-testing] or [junit] or [pytest]"

Adicionando a String, vamos para essa tela:

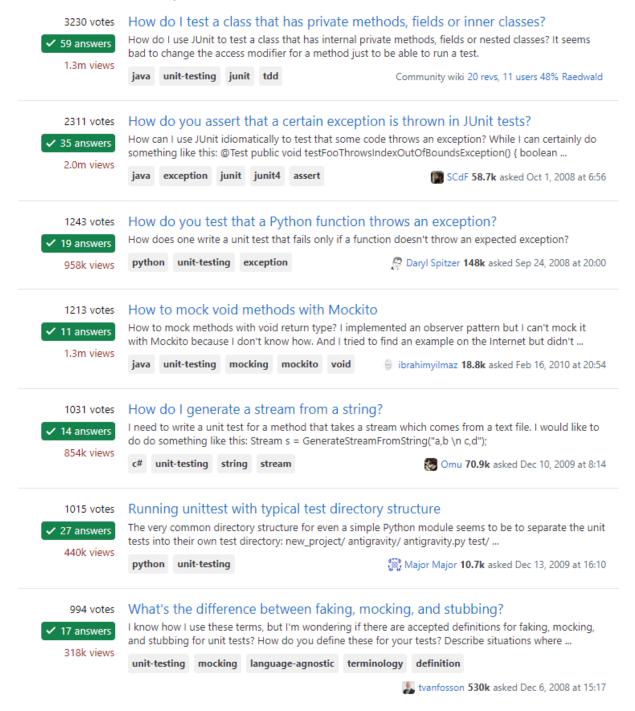


A seguir, clicamos no filtro lateral e acrescentamos o filtro de ordem decrescente de pontuação: "Highest score"



A seguir, selecionamos uma pergunta cadastrada no StackOverflow que tenha ao menos uma resposta aceita e no mínimo 400 votos.

Dentro todas as disponíveis, selecionamos uma:



A selecionada será "How to test the type of a thrown exception in jest"

How to test the type of a thrown exception in Jest

Asked 6 years, 11 months ago Modified 10 months ago Viewed 750k times



Esse foi o Questionamento colocado no StackOverflow por um usuário.

E essa foi a resposta mais avaliada e aceita pelo Site:

```
In Jest you have to pass a function into expect(function).toThrow(<blank or type of error>).

Example:

test("Test description", () => {
    const t = () => {
        throw new TypeError();
    };
    expect(t).toThrow(TypeError);
});

Or if you also want to check for error message:
```

```
test("Test description", () => {
  const t = () => {
    throw new TypeError("UNKNOWN ERROR");
  };
  expect(t).toThrow(TypeError);
  expect(t).toThrow("UNKNOWN ERROR");
});
```

If you need to test an existing function whether it throws with a set of arguments, you have to wrap it inside an anonymous function in expect().

Example:

Parte 2)

Com base nesse questionamento feito por um usuário membro do Stack Overflow, pegamos e criamos um código para que possamos mostrar como solucionar o problema proposto por ele utilizando a resposta mais bem avaliada para aquele questionamento no Stack Overflow

Nosso código foi esse:

```
function throwError() {
   throw new TypeError('This is a tpe error');
}

module.exports = throwError;
```

E a solução para ela utilizando a resposta do usuário do Stack Overflow foi essa:

```
const throwError = require('./exemplo');

test("should throw TypeError", () => {
    const t = () => {
        throwError();
    };
    expect(t).toThrow(TypeError);
    //expect(t).toThrow('This is a type error');
};
```

Trazendo o resultado esperado no Jest do terminal da IDE:

Caso o tipo da exceção lançada não fosse a mesma, daria erro, retornando no comando "npm test" que havia um erro e mostrando onde o erro se encontrava no código para que o usuário possa fazer a devida correção;

Para conseguirmos fazer o teste com o Jest, precisamos inicialmente instalar o Jest em nossa máquina através do terminal.

Abaixo fica todo o processo que precisamos fazer para instalar o Jest:

Iniciando

Instale Jest usando seu gerenciador de pacotes favorito:

npm Yarn pnpm

```
npm install --save-dev jest
```

Vamos começar por escrever um teste para uma função hipotética que soma dois números. Primeiro, crie um arquivo sum. js:

```
function sum(a, b) {
  return a + b;
}
module.exports = sum;
```

Em seguida, crie um arquivo chamado sum.test.js. Este irá conter o nosso teste real:

```
const sum = require('./sum');
test('adds 1 + 2 to equal 3', () => {
   expect(sum(1, 2)).toBe(3);
});
```

Adicione a seguinte seção ao seu package.json:

```
{
   "scripts": {
     "test": "jest"
   }
}
```

Por fim, execute yarn test ou npm run test e o Jest irá imprimir a seguinte mensagem:

```
PASS ./sum.test.js
√ adds 1 + 2 to equal 3 (5ms)
```

Para finalizar, criamos também outro código, referenciando o mesmo questionamento colocado no Stack Overflow sobre "Como testar o tipo de uma exceção lançada por uma função?"

O código criado foi o seguinte:

E o código de teste em Jest para esse código foi esse:

```
▲ geradorErro.test.js > ...

  1 // errorGenerator.test.js
  const errorGenerator = require('./geradorErro');
  4 test('throws TypeError when type is "type"', () => {
      const t = () => {
          errorGenerator('type');
       1:
       expect(t).toThrow(TypeError);
       expect(t).toThrow('This is a type error');
     });
 11
      test('throws ReferenceError when type is "reference"', () => {
 12
 13
      const t = () => {
          errorGenerator('reference');
       };
        expect(t).toThrow(ReferenceError);
      expect(t).toThrow('This is a reference error');
      });
     test('throws generic Error when type is not specified', () => {
       const t = () => {
 21
          errorGenerator('other');
 23
       };
       expect(t).toThrow(Error);
        expect(t).toThrow('This is a generic error');
 26 });
```

Esses testes fazem as seguintes coisas:

Primeiro teste (throws TypeError when type is "type"):

- Verifica se a função **geradorErro** lança um TypeError quando chamada com o argumento 'type'.
- Verifica também a mensagem do erro para garantir que ela é a esperada.

Segundo teste (throws ReferenceError when type is "reference"):

- Verifica se a função geradorErro lança um ReferenceError quando chamada com o argumento 'reference'.
- Verifica também a mensagem do erro para garantir que ela é a esperada.

Terceiro teste (throws generic Error when type is not specified):

- Verifica se a função **geradorErro** lança um erro genérico (Error) quando chamada com qualquer outro argumento.
- Verifica também a mensagem do erro para garantir que ela é a esperada.

Dessa forma, também respondemos o questionamento proposto no Stack OverFlow.

O Repositório do GitHub é o seguinte:

https://github.com/Ian-Silva/Teste Software 2024 Conceicao Ian/tree/main