

Lançando exceções com objetos mocados:

E se precisássemos que um objeto **mockado** tivesse como comportamento lançar uma **exceção**? Podemos fazer isso.

Vamos imaginar **então que o serviço do SPC seja instável e lance uma exceção**, meu **sistema vai ter que pegar essa exceção** e transforma-la em **uma mensagem amigável** quando o serviço do SPC estiver disponível. Vamos usar o **TDD e começar pelo teste**:

```
@Test
public void deveTratarErroNoServicoDoSPC() throws FilmeSemEstoqueException, LocadoraException {
    //GIVEN
    Usuario usuarioTentandoAlugar = UsuarioBuilder.umUsuario().agora();
    List<Filme> filmeParaAlugar = Arrays.asList(FilmeBuilder.umFilme().agora());

    Mockito.when(spcService.possuiNegativacao(usuarioTentandoAlugar))
        .thenReturn(new Exception("Falha catastrófica"));

    //WHEN
    servicoDeLocacao.alugarFilme(usuarioTentandoAlugar, filmeParaAlugar);

    //THEN
}
```

Até aqui temos o **cenário definido**, temos o **usuarioTentandoAlugar** e sua lista de filme **filmeParaAlugar**, como parte do **cenário também temos o comportamento de exceção que QUEREMOS QUE NOSSO MOCK LANCE**. Então quando eu chamar o **spcService.possuiNegativacao()** eu quero que ele me lance uma **nova Exception com a mensagem "falha catastrófica"**.

Certo, agora eu quero definir o **comportamento do meu sistema** para quando eu receber essa **exceção do spcService**, eu quero que **ele segure a exceção** e me **retorne uma mensagem para tentar alugar mais tarde**:

```
    .thenReturn(new Exception("Falha catastrófica"));

    //THEN
    excecao.expect(LocadoraException.class);
    excecao.expectMessage("problemas no SPC, tente novamente.");

    Errors: 1    Failures: 0
```

Obviamente o teste falha(esse é o **objetivo do TDD**), isso porque não alteramos o nosso código. Primeiro que o método de **possuiNegativacao()** na **interface de SPCService não lança exceção nenhuma**, então vamos mudar isso:

```
public boolean possuiNegativacao(Usuario usuario);
```

```
public boolean possuiNegativacao(Usuario usuario) throws Exception;
```

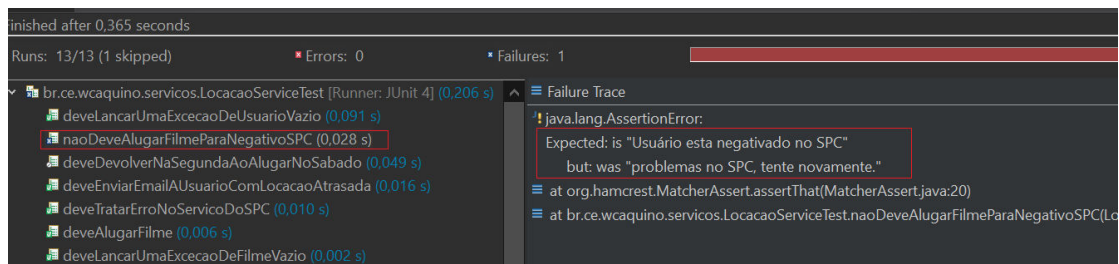
Agora podemos tratar a Exception que o método do spcService lança.

Agora que esse método está lançando uma exceção, precisamos tratar nossa classe de serviço que utiliza esse método. Exatamente nesse trecho não tinha uma tratativa porque não precisava, agora que `spcService` lança uma exceção precisamos inserir um trycatch:

```
if(spcService.possuiNegativacao(usuario)) {  
    throw new LocadoraException("Usuário esta negativado no SPC");  
}
```

```
try {  
    if(spcService.possuiNegativacao(usuario)) {  
        throw new LocadoraException("Usuário esta negativado no SPC");  
    }  
} catch (Exception e) {  
    throw new LocadoraException("problemas no SPC, tente novamente.");  
}
```

Executando todos meus testes, recebo um erro no meu teste `naoDeveAlugarParaUsuarioNegativoSPC()`, era pra eu ter recebido a mensagem "usuario esta negativado no SPC", mas acabei recebendo "problemas no SPC, tente novamente.". Isso provavelmente é um erro de lógica do método, então vamos voltar e refatora-lo(mais um Objetivo do TDD):



O problema está no try catch dessa verificação, lançamos ela no bloco de try, então quando o método encontrou essa exceção ele a capturou no bloco de catch e sobrescreveu a mensagem:

```
try {  
    if(spcService.possuiNegativacao(usuario)) {  
        throw new LocadoraException("Usuário esta negativado no SPC");  
    }  
} catch (Exception e) {  
    throw new LocadoraException("problemas no SPC, tente novamente.");  
}
```

Podemos fazer a refatoração dessa lógica de algumas maneiras. Vamos usar a seguinte lógica.

Vamos criar um booleano para saber se o usuário está ou não negativo no SPC, então na utilização do `spcService` fazemos no try catch que poderá lançar a

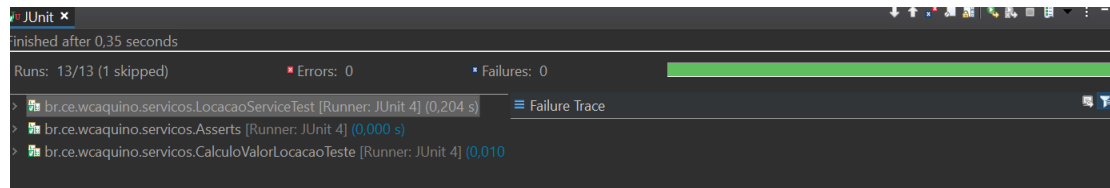
`LocadoraException("Problemas no SPC, tente novamente.")`, e usamos o booleano **retornado** para verificar se a `LocadoraException("Usuario negativado no SPC")` deverá ser lançada:

```
boolean usuarioEstaNegativado;

try {
    usuarioEstaNegativado = spcService.possuiNegativacao(usuario);
} catch (Exception e) {
    throw new LocadoraException("problemas no SPC, tente novamente.");
}

if(usuarioEstaNegativado) {
    throw new LocadoraException("Usuário esta negativado no SPC");
}
```

Com isso todos os testes estão passando:

A screenshot of the JUnit test runner interface. At the top, it says "finished after 0,35 seconds". Below that, it shows "Runs: 13/13 (1 skipped)", "Errors: 0", and "Failures: 0". A green progress bar is visible. The bottom section lists the test classes: "br.ce.wcaquino.servicos.LocacaoServiceTest [Runner: JUnit 4] (0,204 s)", "br.ce.wcaquino.servicos.Asserts [Runner: JUnit 4] (0,000 s)", and "br.ce.wcaquino.servicos.CalculoValorLocacaoTeste [Runner: JUnit 4] (0,010 s)". There is a "Failure Trace" button on the right.

```
JUnit x
finished after 0,35 seconds
Runs: 13/13 (1 skipped) Errors: 0 Failures: 0
> br.ce.wcaquino.servicos.LocacaoServiceTest [Runner: JUnit 4] (0,204 s)
> br.ce.wcaquino.servicos.Asserts [Runner: JUnit 4] (0,000 s)
> br.ce.wcaquino.servicos.CalculoValorLocacaoTeste [Runner: JUnit 4] (0,010 s)
```