# Tratamento de exceção 2:

## Forma elegante:

A forma **"elegante"** de se tratar uma exceção é bem enxuta, porém a utilizamos na versão anterior usando apenas a superclasse "Exception". Com isso não podemos especificar a mensagem da exceção e isso torna o teste um tanto quanto **genérico.**

Para que a forma **"elegante"** tenha segurança é necessário que especifiquemos a classe de exceção a qual aquele teste retornará, portanto, invés de lançarmos uma Exception na verificação do método de alugar um filme, nós criamos uma classe **FilmeSemEstoqueException** que estende **Exception** e o utilizamos na anotação, garantindo assim que a exceção foi bem especifícada. Com isso não é necessário usar os outros dois métodos apresentados para testar exceção, claro que tudo isso é uma questão de gosto e seu modo de organização.

## Forma robusta:

Vamos adicionar mais duas condições na nossa locacaoService, o usuario e nem o filme podem ser nulos, vamos usar uma unica exceção criado para o projeto e fazemos o polimorfismo da mensagem apenas. Nosso método de locação não usa mais uma Exception, agora damos um throw nas exceções criadas.

O método de para o usuário vazio só quer a exceção de LocadoraException, então jogamos a FilmeSemEstoqueException para o JUnit gerenciar. Então envelopamos o método a ser testado com o trycatch pegando usando a asssertiva na mensagem, sem esquecer de usar .fail( ).

## Forma Rule:

Usando a rule temos que jogar as duas exceções para gerenciamento do JUnit, usando a forma "Rule" temos que fazer a validação dentro do cenário do teste (GIVEN/ DADO QUE).

## Resumo:

A forma **"elegante"** funciona muito bem quando você tem uma classe de Exception específica para tratar o problema, ou seja, garantir que a exceção seja lançada apenas por aquele motivo.

A forma **"robusta"** é a que você tem mais poder sobre a execução, por isso o nome, essa forma te da um melhor poder de manipulação sobre a exceção, depois que executa o trycatch o método continua seu fluxo.

Já a forma **"Rule"** tem quase o mesmo comportamento da robusta, como precisar da mensagem, porém ela não continua o fluxo.

Claro que todas elas tem seus pontos positivos e negativos e tudo depende de como você irá usa-las, uns testes podem ser 100% satisfeitos com a forma "elegante" enquanto outros talvez precisem da forma robusta, tudo depende.