

Resumo

Transcrição

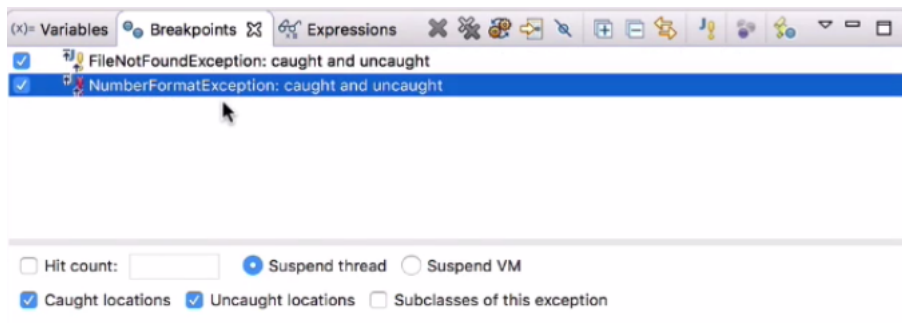
Repassando as aulas vistas até então, vamos relembrar dos *breakpoints* condicionais, sobre os quais vimos dois tipos: um, **condicional**, com a condição pré-estabelecida a ser executada, e o outro, **condicional de exceção**, que pausa durante modo *Debug* somente quando houver alguma exceção assim definida.

Para se definir um *breakpoint* condicional, dependendo da condição desejada, é necessário inicialmente termos um *breakpoint*. Neste caso, na aba `CarrinhoCompra.java`, o acrescentamos à localização do método que calcula o preço final da compra, dentro do referente laço, com a variável `produto`.

Clicando com o botão direito do mouse em cima do *breakpoint* criado anteriormente, selecionamos a opção "*Breakpoint Properties*", marcando o *checkbox* de "*Conditional*" na nova janela que se abriu, digitando `produto.getDescricao().equals(*Cadeiras*)`, em que `Cadeiras` representa a condição a ser seguida para o funcionamento deste *breakpoint*.

Para retirar o *breakpoint*, basta clicar duas vezes em cima da bolinha ao lado da linha de código. Abre-se uma janela pop-up perguntando se queremos realmente tirá-lo dali, algo que não ocorre quando queremos deletar *breakpoints* simples.

Para colocar um *breakpoint* em uma exceção específica, mudamos para a perspectiva *Debug*, que permite a visualização da aba "*Breakpoints*". Quando um arquivo não é encontrado, por exemplo, podemos clicar em "*Add Java Exception Breakpoint*", digitando `FileNotFoundException`. Então, ele capturará estas duas exceções, não havendo limite de quantidade:



Desta forma, aprendemos a colocar uma exceção dependendo de algum código próprio, de qualquer variável desejado, e também a utilizar *breakpoints* condicionados à uma exceção específica.