



Lidando com valores que podem ser null

Como vimos anteriormente, o operador `!!` permite a execução de variáveis que podem ser nulas, porém, ao mesmo tempo é muito arriscado, pois ele pode lançar a famosa NPE (`NullPointerException`).

Considerando esse detalhe, vamos praticar algumas alternativas para lidar com esse tipo de situação, ou seja, garantir que o valor não é `null` .

Protegendo o código com o if

Uma das alternativas iniciais é aplicar a estrutura de `if` assim como fazemos também no Java. Portanto, antes de realizar alguma operação com a `property view` , adicione um `if` em volta dela que verifica se ela não tem o valor `null` .

Protegendo com a Safe Call

Uma outra alternativa é utilizando a *Safe Call*, ou seja, utilize o operador `?.` antes de chamar um membro da `property view` .

Lembrando que essa abordagem não permite o uso do `with` do jeito que ele está implementado.

Protegendo com a Safe Call junto da função let

A última alternativa que vimos é justamente a chamada da *Safe Call* com a função `let` , portanto, se optar por essa chamada, é basicamente o mesmo código feito no `if` , a diferença é que primeiro faz a *Safe Call* e depois chama a função `let` .

Assim que abrir o escopo do `let` a `view` será considerada como uma `view` que não pode receber um valor `null` .

Testando o código

Escolha uma das abordagens e aplique em cada um dos pontos do código que não compila devido ao uso de uma variável que pode receber um valor `null` . Após aplicar, execute a App sem inicializar a `property viewDaActivity` e veja o que acontece na App.