02

Extração

Transcrição

[00:00] Implementei essa lógica, mas ainda está bem feio. Vamos extrair esse código. Vamos remover desse mapa, nessa posição, quatro casas. Coloco a quantidade, deixo isso bonito.

[01:15] Extraímos a função, mas será que é suficiente? Não, a função ainda está bem confusa. Ela usa uma variável direita para ficar somando um número cada vez maior. Calma lá. Quero limpar o que está perto de mim. Imagina a complexidade daqui a pouco.

[01:36] Vamos refatorar mais essa função. Primeiro, esse contexto de remover não tem nada a ver com nova. É simplesmente uma posição. Remove dessa posição quem estiver lá. Removemos os quatro, mas toda vez vai ficar somando? Meio estranho. Ao invés disso, eu poderia andar para a direita. A posição é o meu herói, ele sabe andar para a direita, então calculo nova posição.

[02:22] Ao invés de ficar usando direita, eu poderia falar que a posição é igual a posição, calcula nova posição, para a direita, ou seja, a letra D. Ele sabe andar para a direita, não preciso mais ficar somando.

[02:54] Extraímos a função, mapa, a posição a partir da qual vamos somar um, dois, três, quatro, até o quatro fazemos um laço, depois dizemos para andar uma posição para a direita e limpar. E não só isso. Essa parte de limpar também já existe. A própria posição já tem um método remove do mapa, que limpa ele do mapa. Usamos duas funções que já existiam.

[03:41] Nosso código ainda está um pouco estranho. De onde surgiu o D? O que quero fazer é andar para a direita. Se você vai invocar um método chamado direita, preciso criar o método que faz calcula nova posição com o valor D. Ele vai retornar esse valor. Por padrão é o return. O código fica bem mais simples.

[04:33] Remover um cara do mapa por causa da bomba significa ir de um até o número de casas que você quer andar, anda para a direita e remove. É isso que fizemos.