

## Elementos

### Transcrição

[00:00] Comecei falando que eu ia ensinar uma estrutura, um agrupamento de dados. E de repente no meio da história coloquei alguns comportamentos. Em programação, em geral, quando aprendemos o básico, aprendemos estrutura de dados, que é o agrupamento de dados. Se agrupa valores ali dentro. Já em uma classe, vamos agrupar os valores e os comportamentos ligados a esses valores. Temos não só a linha e a coluna, como também os comportamentos que estão ligados a eles, que representam o herói e estão todos dentro do herói. Nesse caso, o herói é uma classe, é uma característica, que nem um jogo de RPG. Mas eu sou o herói. Assim como tenho a classe elfo e o personagem de O Senhor dos Anéis. E aí posso ter vários heróis, cada um em uma linha e coluna diferente. Mas todos sabem se comportar como herói.

[01:20] A diferença entre uma estrutura de dados tradicional para a orientação a objetos, instâncias, valores para esses heróis, é que aqui estamos adicionando comportamentos a essas estruturas, não estamos ficando só com os dados. E a partir daí surge uma sequência de características interessantes das quais podemos nos beneficiar quando aprendemos mais de orientação de objetos. Nosso foco aqui no curso, por mais que seja só ensinar estrutura, acabei ensinando um pouco do básico de como criar uma classe, ou seja, agrupar tanto os comportamentos quanto os atributos, instanciar objetos, valores dessa classe, e trabalhar com eles invocando os métodos.

[02:06] Eu fui um pouco além do necessário, mas é interessante para nós, porque em Ruby é muito comum trabalhar com classes de verdade, atributos e comportamentos juntos, e não só com estruturas e valores que não fazem nada lá dentro. Muitas vezes chamamos essas classes que só tem os valores e não fazem nada de classes anêmicas. Elas não sabem se comportar.

[02:38] No nosso caso, nossa classe não é anêmica, ela tem valores e atributos e faz diversas coisas. Essas funções que só podem ser invocadas se você tem um herói na sua mão chamamos de métodos. Temos os atributos e os métodos, que são os comportamentos que vão formar nossa classe de herói. Já tínhamos visto diversos desse tipo antes. Toda vez que trabalhávamos com string, que era uma classe, tinha métodos da string que invocamos. Toda vez que trabalhamos com números, invocamos métodos, toda vez que trabalhamos com arrays invocamos métodos.

[03:20] Em Ruby, tudo vai ser basicamente um objeto e vai ter métodos para chamar. Por isso mesmo não quis ficar só na estrutura básica. Quis mostrar que conseguimos colocar comportamentos lá dentro da classe e ela tem tantos dados quanto comportamentos, formando uma classe que podemos instanciar, criar objetos daqueles tipos.