02

Introdução ao REST

Transcrição

[00:00] Nós vamos desenvolver nesse capítulo uma funcionalidade usando essa tecnologia REST, que é utilizada quando queremos integrar aplicações. Eu tenho uma aplicação *mobile* para se comunicar com aplicação em Java no back-end. Como eu faço para essas duas aplicações compartilharem informação uma com a outra?

[00:21] O que vamos ver é que vamos desenvolver uma aplicação no *browser* do usuário usando uma tecnologia chamada Vue, que vai ser explicada posteriormente. Essa aplicação precisa buscar dados de pedidos para poder criar a página de ofertas e precisa também uma maneira de criar ofertas, de enviar uma requisição para a nossa aplicação, para criar ofertas aos pedidos.

[00:44] E vamos utilizar o REST, como eu falei. Como funciona o REST? O objetivo dele é simplificar essa interação entre as aplicações para que não tenhamos dificuldade de integrarmos aplicações em linguagens e tecnologias diferentes. Para isso ele utiliza o próprio HTTP. Usando HTTP ele organiza essa comunicação entre a aplicação. Como ele faz isso?

[01:10] Imagine que existe uma aplicação rodando no *browser* do usuário com JavaScript e essa aplicação precisa se comunicar com a aplicação Mudi, que é feita em Java, que está rodando no servidor. E ela precisa buscar pedidos para poder listar pedidos para o usuário, que é exatamente a tela que vamos fazer. Nós listamos ali pedidos para poder o usuário gerar ofertas para esses pedidos.

[01:38] O que essa aplicação nossa que está rodando no *browser* vai fazer? Ela vai fazer uma requisição HTTP para "/pedidos", que é uma requisição usando o

verbo GET. Essa requisição bate na aplicação Mudi, a aplicação vai processar essa requisição "/pedidos" e o processamento é basicamente ir no banco de dados, buscar informações de uma lista de pedidos e devolver isso para o *browser*.

[02:02] Então a nossa aplicação Mudi não vai gerar mais um HTML com os pedidos que vão ser gerados oferta e tal, ela vai fornecer os dados de pedidos para que uma aplicação rodando no *browser* gere esse HTML. Estamos falando uma aplicação de *front-end* se comunicando com a nossa aplicação de back-end, que é o que viemos desenvolvendo até agora.

[02:26] Como não é uma página HTML que vai ser renderizada pelo *browser*, esse retorno vai ser feito transformando essa lista de pedidos que veio do banco de dados em uma lista de pedidos em JSON. Ele vai transformar essa lista. Como o JSON funciona?

[02:47] O JSON basicamente tem esse abre e fecha chaves, quando você vai declarar um determinado objeto, que você vai escrever ali. Aqui eu estou dizendo que ele tem um atributo chamado de "pedidos" e esse "pedidos" é um *array*. Você vê que você que isso é um abre colchetes, que fecha colchetes aqui embaixo.

```
{
    "pedidos": [
          {
            "Id": 123
            ...
           },
          ...
     ]
}
```

COPIAR CÓDIGO

[03:07] E dentro desse abre e fecha colchetes representa um *array*, ele vai ter então um *array* de objetos separado por vírgula e cada objeto desse tem os dados de um pedido. Aqui eu coloquei só o "Id" representando, mas esses três pontos representam o resto dos atributos: nome do produto, "urlProduto", "urlImagem" e "descricao" (do pedido); são os atributos do pedido.

[03:33] E essa vírgula representa que depois daqui, `},` , vem novos objetos de pedido em sequência, fechando então o *array* de pedidos.

[03:43] Veja que essa requisição de pedidos que fazemos para o Mudi, ela não vai retornar um HMTL gerado pelo Thymeleaf, mas vai retornar um JSON com os dados de um determinado estado daquela lista de pedidos que estão no banco de dados.

[03:59] Usando o REST, para você buscar informações, você usa a tecnologia do HTTP, de url, de verbo HTTP, de código 200 informando que foi processado com sucesso, e você retorna um objeto JSON.

[04:20] E temos outras operações que podemos fazer. Por exemplo: se você quiser criar um determinado pedido, veja que aqui no GET você apenas faz uma requisição pedindo os pedidos e ele retorna o JSON.

[04:33] Aqui na criação de um pedido nós mandamos um JSON – podem ser outros formatos, é o que vamos utilizar nesse treinamento o JSON porque é o mais utilizado – nós mandamos as informações de um determinado pedido, com nome do produto, "urlProduto", "urlImagem" e "descricao", de novo. Esses três pontos representam que alguma coisa aqui foi preenchida pelo usuário.

```
{
  "nomeProduto":"...",
  "urlProduto":"...",
  "urlImagem":"...",
  "descricao":"..."
}
```

COPIAR CÓDIGO

[04:55] Isso vai como um JSON nessa requisição POST, que é recebida pela aplicação em Java, que vai salvar isso no banco de dados de pedidos e vai retornar um código 200 (se tiver dado tudo certo) com, por exemplo, o ID desse novo pedido que foi criado, "124", por exemplo.

[05:16] Só que não adianta eu só conseguir buscar uma lista de pedidos e eu criar pedido, eu também preciso, por exemplo, buscar um determinado pedido, e voltar essa requisição onde não passamos nada, como o GET aqui de cima, GET "/pedidos". Então fazemos uma requisição GET para "/pedidos/" o ID do pedido que queremos buscar a informação.

```
{
  "id":124,
  "nomeProduto":"...",
  "urlProduto":"...",
  "urlImagem":"...",
  "descricao":"..."
}
```

COPIAR CÓDIGO

[05:40] Aí a nossa aplicação em Java vai no banco de dados e retorna o JSON – código "200" de novo do processo – com toda a informação, incluindo o pedido. Veja que no POST não tem ID, porque o ID é gerado no banco de dados, do nosso caso aqui.

[05:57] Então já sabemos buscar uma lista de pedidos, criar um determinado pedido, pegar um determinado pedido e como atualizamos um determinado pedido. Por exemplo: eu quero atualizar a descrição de um determinado pedido, isso nós fazemos através de uma requisição para "/pedidos/" o ID do pedido que queremos atualizar - e essa requisição é do tipo PUT.

```
{
  "descricao":"..."
}
```

COPIAR CÓDIGO

[06:21] E veja que vai uma informação que é só a descrição, eu quero atualizar a descrição do pedido com o ID 124, verbo PUT do HTTP. Ele atualiza no banco de dados e retorna um "200". Foi atualizado com sucesso!

[06:40] A proposta do REST é utilizar o HTTP para conseguirmos trocar informações entre sistemas, sem que eles precisem utilizar alguma tecnologia nova e tudo mais. Usamos o próprio HTTP para isso.

[06:53] Então é isso. Até o próximo vídeo!