09

HTTP2 - Multiplexação

Transcrição

Outra coisa importante de requisição é que temos o conceito de `request` e `response`. Cada requisição e cada resposta no HTTP1.1 são únicos.

"Por baixo dos panos", antes dessa requisição de fato ser feita, há uma conexão, comunicação entre cliente e servidor, que chamamos de **TCP**. Para que consigamos realizar uma requisição via HTTP, antes existe um modelo de TCP, que é um protocolo de transporte. Isso é mais a nível de infraestrutura, pois quando trabalhamos com desenvolvimento, acabamos deixando isso pra lá, já que ficamos na camada acima dessa conexão.

Queremos mostrar é que quando fazemos uma requisição, ela é única. No HTTP, cada requisição deveria abrir uma conexão TCP, executar e fechar.

Mas isso seria muito ruim porque conexão TCP é recurso caro, é um recurso que demora a ser alocado. Claro que é muito rápido a nível computacional, mas é mais um passo antes da requisição HTTP prosseguir e recebermos uma resposta.

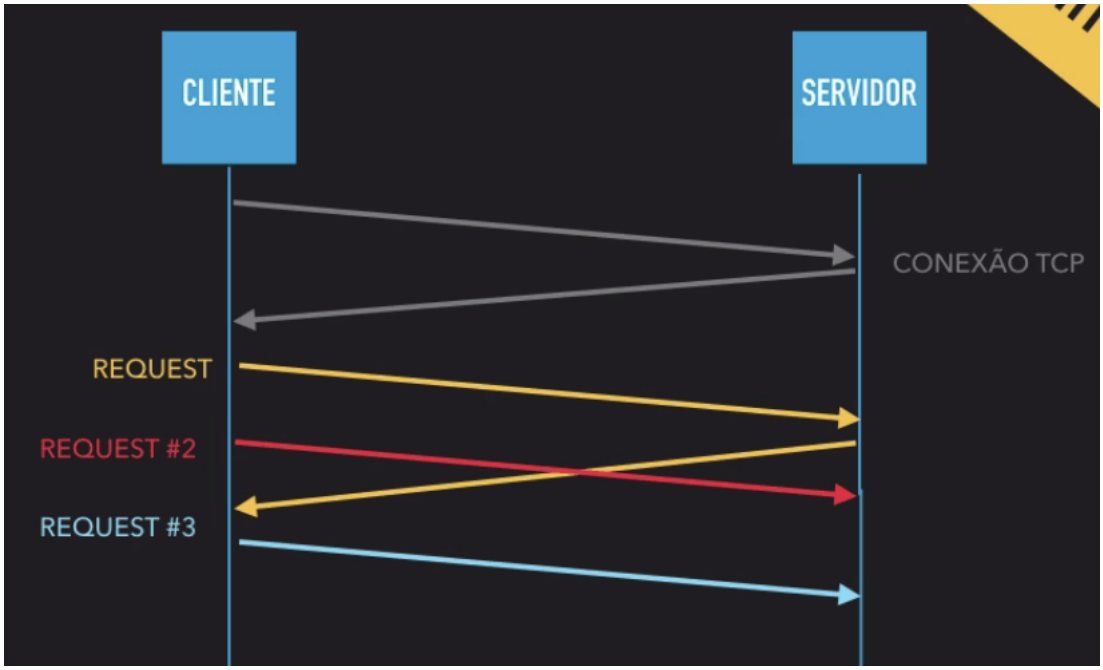
Então o que acontece, no HTTP1 existe um mecanismo chamado de `Keep-Alive`. O `Keep-Alive` determina quanto tempo, por exemplo, a nossa conexão pode ficar ativa. Ou seja, não encerra essa conexão TCP. Portanto, conseguimos realizar várias requisições com a mesma conexão TCP.

Hoje, na maioria dos browsers, temos um número entre 4 e 8 de conexões simultâneas por domínio. Significa que se fizermos uma requisição para a página da Caelum e a página da Caelum tiver mil recursos, o browser tem 4 a 8 conexões TCP ativas para conseguir realizar essas requisições em paralelo, e não serial. Mas isso na versão 1.1.

Keep-Alive no HTTP2

O `Keep-Alive` continua existindo no HTTP2, só que ele trouxe uma novidade. Por exemplo, se temos uma conexão TCP aberta e realizamos uma requisição, poderíamos já dar prosseguimento às próximas requisições, isso em paralelo, sem de fato ficar esperando o resultado dela, de maneira assíncrona, e vamos recebendo essas respostas à medida em que o servidor for conseguindo processar.

Na imagem abaixo, fizemos a requisição 1 e requisição 2, quando íamos fazer requisição 3, já recebemos uma resposta:



Então, essas requisições e respostas vão chegando a todo tempo. É totalmente paralelo. A mesma coisa acontece com o servidor, não precisamos esperar uma resposta para enviar outra. Se já está pronta para ser enviada, ele já envia diretamente.

Esse conceito que surgiu no HTTP2 é chamado de ***Multiplexing*** e traz uma performance bastante relevante para o nosso HTTP.