

Utilizando filtros e logging

A gente já sabe fazer `GET` , `POST` , `DELETE` , entende os response codes, sabe que existe cabeçalhos indo para o servidor, cabeçalhos que estão indo do servidor para o cliente.

Legal. Agora a gente está usando bastante o `curl` , certo? E se dá de repente um probleminha bem leve no meu servidor e eu quero ver o que é que meu cliente está comunicando com meu servidor? Então eu posso ver o que é que o meu cliente JAX-RS está enviando para o servidor e o que é que o servidor está devolvendo para o meu cliente configurando o meu cliente.

Então quando eu pego aqui esse

```
this.client = ClientBuilder.newClient();
```

ao invés de criar um cliente padrão, eu vou criar a configuração de um cliente.

```
ClientConfig config = new ClientConfig();
```

Agora que eu tenho essa configuração do meu cliente eu vou registrar uma *API* de log, que é o `LoggingFilter` .

```
config.register(new LoggingFilter());
```

O `LoggingFilter` é do *Jersey*, que é uma implementação do *JAX-RS*. Então, dado o `LoggingFilter` , eu crio agora um `ClientBuilder.newClient` baseado nessa configuração. Legal?

```
this.client = ClientBuilder.newClient(config);
```

Então, eu posso criar um cliente com algumas configurações extras. Essas configurações podem registrar diversos tipos de filtros. O filtro que eu vou usar aqui agora é o filtro de `log` do *Jersey*.

Vamos ver qual é o resultado? Vou rodar meus testes. Venho agora aqui no *console* e olha só, durante os testes ele imprimiu : "Estou dando um `GET` para o `http://localhost:8080/carrinhos/1` . O servidor devolveu 200 com esses cabeçalhos. Depois, no outro teste, eu fiz um `POST` para `http://localhost:8080/carrinhos/1` . O servidor devolveu 201 com esse location [`http://localhost:8080/carrinhos/1`]. Aí eu fiz um `GET` para esse location e o servidor devolveu 200 com o conteúdo."

Legal, então os filtros são uma maneira de alterar o comportamento do meu cliente. No meu caso eu estou usando um filtro bem básico e interessante, que a gente acaba usando na prática pra debugar o que está acontecendo de um lado para outro, que é o `LoggingFilter` do *JAX-RS*.

Vamos lá, utilizar o `LoggingFilter` e ver o que acontece com o nosso código?

