

## Método Post

### Transcrição

Agora vamos simplificar o método `post()`, localizado dentro do `HttpService.js`:

```
post(url, dado) {  
  
  return new Promise((resolve, reject) => {  
  
    let xhr = new XMLHttpRequest();  
    xhr.open("POST", url, true);  
    xhr.setRequestHeader("Content-type", "application/json");  
    xhr.onreadystatechange = () => {  
  
      if (xhr.readyState == 4) {  
  
        if (xhr.status == 200) {  
  
          resolve(JSON.parse(xhr.responseText));  
        } else {  
  
          reject(xhr.responseText);  
        }  
      }  
    };  
    xhr.send(JSON.stringify(dado));  
  });  
}
```

Vamos reescrevê-lo, adicionando o `fetch`. Como parâmetros usaremos a `url` e uma configuração da requisição que será recebida.

```
post(url, dado) {  
  
  return fetch(url, {  
    headers: { 'Content-Type': 'application/json' },  
    method: 'post',  
    body: JSON.stringify(dado)  
  })  
}
```

No `headers`, adicionando dentro das chaves, um objeto JavaScript (`Content-type`) e definimos seu valor. Em seguida, no `body`, converteremos o dado enviado de JSON para String.

Vamos também lidar com a parte de erro.

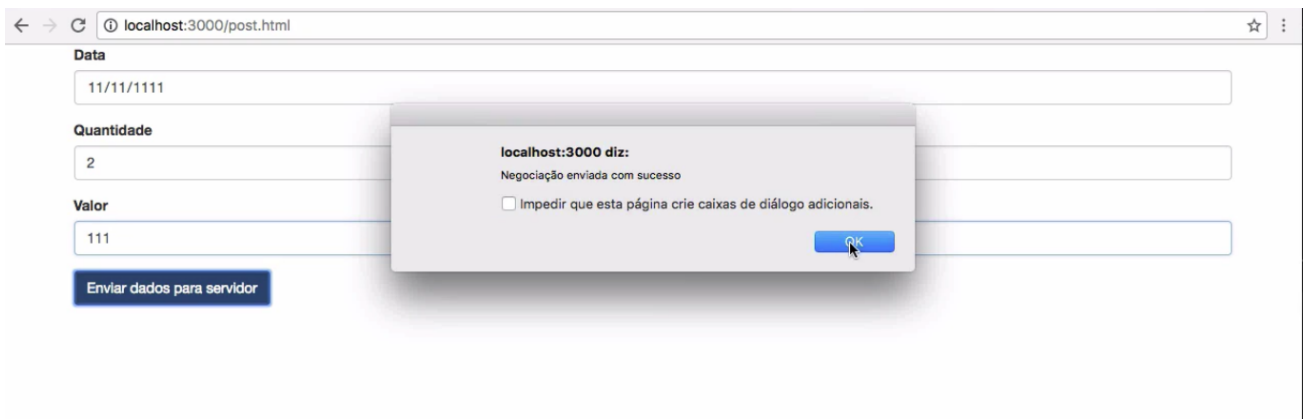
```
post(url, dado) {  
  
  return fetch(url, {  
    headers: { 'Content-Type': 'application/json' },
```

```
        method: 'post',
        body: JSON.stringify(dado)
    })
    .then(res => this._handleErrors(res));
}
```

No caso de erro, será lançada uma exceção e quem estiver usando o `post` do `Http` e chamar o método `catch()`, receberá a mensagem de erro. Em `post.html`, veremos o método sendo chamado:

```
new HttpService()
    .post('/negociacoes', negociacao)
    .then(() => {
        inputData.value = '';
        inputQuantidade.value = 1;
        inputValor.value = 0.0;
        inputData.focus();
        alert('Negociação enviada com sucesso');
    })
    .catch(erro => alert('Não foi possível enviar a negociação: ${(erro)`}));
}
```

Voltaremos no navegador e acessaremos `http://localhost:3000/post.html`. Ao adicionarmos uma nova negociação e confirmarmos o envio dos dados, receberemos logo em seguida uma mensagem:



A negociação foi enviada com sucesso. Quando voltarmos para a página principal, veremos que os dados foram adicionados corretamente na tabela.

Conseguimos fechar a Fetch API, que possui ainda outros recursos, mas já podemos utilizar o que já foi visto até aqui. A seguir, veremos se temos como garantir o bom funcionamento do código em todos os navegadores.