

**De:** [lucascassaro@gmail.com](mailto:lucascassaro@gmail.com)

**Para:** Equipe de Desenvolvimento

**Assunto:** Implementação de Processamento Assíncrono com XMLHttpRequest

Olá equipe,

Espero que estejam bem! Como parte do nosso novo projeto de implementação do processamento assíncrono de dados na página principal do e-commerce, vou explicar como o objeto **XMLHttpRequest** será utilizado para realizar as requisições às APIs REST. Vamos seguir o passo a passo abaixo:

### Passo 1: Criação do objeto XMLHttpRequest

O primeiro passo é criar uma instância do objeto **XMLHttpRequest**. Esse objeto é responsável por fazer a comunicação entre o navegador e o servidor de forma assíncrona.

```
javascript
const xhr = new XMLHttpRequest();
```

### Passo 2: Configuração da requisição

Após criar o objeto, precisamos configurar a requisição. Para isso, utilizamos o método `open()`, que recebe três parâmetros principais:

1. O método HTTP (por exemplo, **GET**, **POST**, **PUT**, **DELETE**).
2. A URL da API que será chamada.
3. Um booleano que indica se a requisição será assíncrona (**true**).

```
javascript
xhr.open('GET', 'https://api.exemplo.com/produtos', true);
```

### Passo 3: Definição do tratamento da resposta

Precisamos definir uma função que será executada quando a resposta da requisição for recebida. Isso é feito através da propriedade **onreadystatechange**. A função será chamada sempre que o estado da requisição mudar.

```
javascript
xhr.onreadystatechange = function() {
  if (xhr.readyState === 4 && xhr.status === 200) {
    // A requisição foi concluída com sucesso
    const resposta = JSON.parse(xhr.responseText);
    console.log(resposta); // Processar a resposta aqui
  }
}
```

```
}  
};
```

- **readyState === 4**: Indica que a operação foi concluída.
- **status === 200**: Indica que a requisição foi bem-sucedida (código HTTP 200 OK).

#### Passo 4: Envio da requisição

Por fim, enviamos a requisição utilizando o método **send()**. Se for uma requisição **POST**, por exemplo, podemos enviar dados no corpo da requisição.

```
javascript  
xhr.send();
```

#### Ciclo completo:

1. O usuário realiza uma ação na página (por exemplo, clica em um botão "Carregar produtos").
2. O evento dispara a função que cria e configura o **XMLHttpRequest**.
3. A requisição é enviada ao servidor.
4. O servidor processa a requisição e retorna uma resposta.
5. A função **onreadystatechange** processa a resposta e atualiza a interface do usuário.

Qualquer dúvida, estou à disposição para esclarecer. Vamos trabalhar juntos para garantir uma implementação eficiente!

Abraços,

**Lucas Lordes Cassaro da Costa**

Arquiteto de Interoperabilidade Web