

# Requisições AJAX

Prof. Daniel Di Domenico



# Objetivo da aula

- Implementar requisições AJAX:
  - Comunicação entre o cliente (navegador) e o servidor (servidor Web/PHP)
    - Utilização da linguagem JavaScript
- Entender as formas de implementar uma requisição AJAX:
  - Síncrona ou assíncrona
  - GET ou POST
  - JSON

# AJAX

- **Asynchronous JavaScript And XML**
  - Técnica de desenvolvimento para WEB que permite **enviar requisições HTTP ao servidor em segundo plano**
    - Não é uma linguagem de programação
    - Utiliza JavaScript
- **AJAX permite:**
  - Atualizar o HTML da página **sem recarregá-la**
  - Comunicação com o servidor após recarregar a página
  - Busca e envio de dados para o servidor WEB em segundo plano
    - Modo síncrono ou assíncrono
- [https://www.w3schools.com/js/js\\_ajax\\_intro.asp](https://www.w3schools.com/js/js_ajax_intro.asp)

# XMLHttpRequest

- **XMLHttpRequest** é a classe JavaScript que permite a comunicação com um servidor WEB em segundo plano
  - Classe que operacionaliza o AJAX
- Um objeto XMLHttpRequest permite:
  - **1-** Criar uma requisição ao servidor utilizando os métodos HTTP **GET** e **POST** (além dos outros existentes)
  - **2-** Definir se a requisição será enviada de forma síncrona ou assíncrona
    - Se assíncrona, é necessário definir uma função de retorno (callback)
  - **3-** Receber a resposta da requisição em dois formatos:
    - Texto (pode ser um JSON)
    - XML

# Requisição síncrona

- Exemplo de **requisição síncrona**:

```
var url = 'servico.php';  
var xhttp = new XMLHttpRequest();  
xhttp.open('POST', url, false);  
  
xhttp.send();  
  
var retornoTexto = xhttp.responseText;  
//var retornoXML = xhttp.responseXML;  
console.log(retornoTexto);
```

**false:** requisição síncrona (aguarda a resposta do servidor)

# Requisição assíncrona

- Exemplo de **requisição assíncrona**:

```
var url = 'servico.php';  
var xhttp = new XMLHttpRequest();  
xhttp.open('POST', url, true);  
  
xhttp.onload = function() {  
    //Executado após a resposta do servidor  
    var retornoTexto = xhttp.responseText;  
    console.log(retornoTexto);  
}  
  
xhttp.send();
```

**true:** requisição  
assíncrona

Este código só será  
executado quando for  
recebida a resposta do  
servidor

# Envio de dados GET e POST

- Exemplos **GET** e **POST**:

```
//GET
```

```
var url = 'servico_get.php?par=1';  
var xhttp = new XMLHttpRequest();  
xhttp.open('GET', url, false);
```

```
xhttp.send();
```

```
var retorno = xhttp.responseText;  
console.log(retorno);
```

```
//POST
```

```
var dados = new FormData();  
dados.append("par", 1);
```

```
var url = 'servico_post.php';  
var xhttp = new XMLHttpRequest();  
xhttp.open('POST', url, false);
```

```
xhttp.send(dados);
```

```
var retorno = xhttp.responseText;  
console.log(retorno);
```

# JSON

- **JSON:**
  - *JavaScript Object Notation*
  - Formato de troca de dados e informações entre sistemas
    - Descreve **atributos** e **valores**
      - Formato: “chave”: valor
  - Exemplo:

```
{  
  "id":1,  
  "nome":"Fulano de Tal",  
  "endereco":"R. Qualquer"  
}
```





# Recebimento de dados: JSON

- Exemplo de requisição AJAX **recebendo** dados no formato JSON:

```
var url = 'pagina.php';  
var xhttp = new XMLHttpRequest();  
xhttp.open('GET', url, true);  
  
xhttp.send();  
  
var objeto = JSON.parse(xhttp.responseText);
```

O retorno JSON  
(texto) foi convertido  
para um objeto  
JavaScript

# Envio de dados: JSON

- Exemplo de requisição AJAX **enviando dados** no formato JSON:

```
var objetoJson = {  
    "valor": 1  
}  
  
var url = 'pagina.php';  
var xhttp = new XMLHttpRequest();  
xhttp.open('POST', url, true);  
xhttp.setRequestHeader('Content-type',  
    'application/json');  
  
xhttp.send(JSON.stringify(objetoJson));  
  
var retornoTexto = xhttp.responseText;  
console.log(retornoTexto);
```

Objeto JavaScript  
convertido para  
envio na requisição  
como JSON (texto)