Formulários / Envio de dados / PHP

Professor: Luiz Carlos Felix Carvalho

Já vimos como configurar um formulário em HTML para enviar uma requisição HTTP do tipo GET / POST. Veja abaixo, nas Figuras 1 e 2:

```html
<html>
    <head>
        <meta charset="UTF-8">
    </head>
    <body>
        <h1>Nossa primeira submissão para o PHP</h1>
        <form method='GET' action='./message.php'>
            <label for='txt_mensagem' > Mensagem </label><br>
            <input type='text' name='txt_mensagem' id='txt_mensagem' /> <br><br>
            <input type='submit' value='Enviar' /> <br>
        </form>
    </body>
</html>
```

Figura 1: Código HTML para enviar uma requisição do tipo GET

```php
<?php

$message = $_GET['txt_mensagem'];

echo 'mensagem recebida: '. $message;

?>
```

Figura 2: Código PHP para receber requisição do tipo GET

A Figura 1 mostra um código HTML enviando uma requisição HTTP do tipo GET para o endereço './message.php'. A Figura 2 mostra o código que está no arquivo message.php. Ambos estão no mesmo diretório (um qualquer), no servidor Apache (htdocs – por padrão). O código PHP receberá a requisição e imprimirá a mensagem (digitada no campo input do HTML) na tela.

Agora, utilizaremos **requisições HTTP via JavaScript/Ajax** (Asynchronous JavaScript And

XML).

A utilização de requisições assíncronas para comunicação entre o cliente e o servidor é o padrão utilizado hoje em dia. Assim, veremos como realizar uma requisição HTTP JavaScript/Ajax com o método **fetch**.

A fim de melhor exibir o conteúdo dos arquivos utilizaremos o arquivo HTML para armazenar nosso código JavaScript, porém, a melhor prática é separar os arquivos.

Continuemos com nosso formulário que envia uma mensagem para o servidor:

```
<!DOCTYPE html>
<html lang="en-US">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>Formulário - Aula 8! JavaScript e PHP </title>
</head>

<body>
    <h1>Nossa primeira submissão para o PHP com JS</h1>
    <form>
        <label for='txt_mensagem' > Mensagem </label><br>
        <input type='text' name='txt_mensagem' id='txt_mensagem' /> <br><br>
        <input type="button" onclick="sendMessage()" value="Enviar"> </input><br>
    </form>
</body>
</body>

</html>
```

Note que trocamos o input **type submit** pelo input **type button** e que estamos utilizando a propriedade **onclick**, para adicionar um evento a ser executado quando o botão for clicado. O evento informado é um método, **sendMessage()**. Trata-se de um método JavaScript. Assim, vamos definir tal método:

```html
<!DOCTYPE html>
<html lang="en-US">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>Formulário - Aula 8! JavaScript e PHP </title>

    <script>
        function sendMessage() {

            var text = document.getElementById( 'txt_mensagem' ).value;

            return ;
        }

    </script>
</head>

<body>
    <h1>Nossa primeira submissão para o PHP com JS</h1>
    <form>
        <label for='txt_mensagem' > Mensagem </label><br>
        <input type='text' name='txt_mensagem' id='txt_mensagem' /> <br><br>
        <input type="button" onclick="sendMessage()" value="Enviar"> </input><br>
    </form>
</body>
</body>

</html>
```

```php
<?php

$message = $_GET['txt_mensagem'];

echo 'mensagem recebida: '. $message;

?>
```

Note que, na linha 9, criamos nossa área de script para criarmos nosso código JavaScript e definimos o método sendMessage(), já com o código que acessa o valor do input de **id = 'txt_message'.** Basta, agora, criarmos a requisição HTTP para o código PHP. Utilizaremos o método fetch, como já citado. Veja o exemplo abaixo de tal método:

```
fetch( 'http://localhost/nossa-app/getCliente?id=1' );
```

Nesse caso, será enviado uma requisição do tipo GET para o endereço de parâmetro. O método fecth retorna um objeto Promise. Esse objeto representa: “a eventual conclusão

(ou falha) de uma operação assíncrona e seu valor resultante" (developer.mozilla.org). Assim, para tratar o retorno do método fetch, utilizamos os métodos then() e catch() (métodos de Promise). Segue exemplo:

```
fetch('http://localhost/nossa-app/getCliente?id=1')
    .then(data => {
        return data.json();
    }
)
```

Assim, vamos criar nossa chamada HTTP para nosso código PHP:

```
9   <script>
10
11      function sendMessage() {
12
13          var text = document.getElementById( 'txt_mensagem' ).value;
14
15          fetch( './message.php?txt_mensagem=' + text )
16              .then( function ( response ) { return response.json() } )
17              .then( function( dataJson ) {
18
19                  console.info( dataJson );
20
21                  if ( dataJson ) {
22
23                      alert( 'Requisição efetuada com sucesso: ' + dataJson.data );
24                  }
25                  else {
26                      alert( 'Problema!' );
27                  }
28              } )
29              .catch( function( error ) {
30
31                  alert( 'Problema!' );
32              } );
33          return ;
34      }
35
36  </script>
```

Foi criado uma chamada do método fetch passando a url de parâmetro './message.php?txt_mensagem=' com o texto do input concatenado (linha 15). Na linha 16, utilizamos o método then() para informar que, ao retornar a requisição http, queremos acessar o conteúdo de dados da response através do método response.json(). Tal método retorna também uma Promise. Assim, na linha 17 utilizamos o método then para que, ao converter o conteúdo para json executar as linhas de 19 a 27, que apenas imprime na tela um dado enviado pelo serviço PHP. Ainda, na linha 29, temos o método catch que é acionado para os casos em que, a resposta retornada pelo serviço PHP, não conseguir ser convertida para um Json. Assim, se algum problema for retornado pelo PHP, esse código

será executado.

Nosso código PHP, para retornar um Json e a requisição acontecer com sucesso, é o seguinte:

message.php

```php
<?php

    if ( isset ( $_GET['txt_mensagem'] ) ) {

        $message = $_GET['txt_mensagem'];

        echo '{"data": "teste"}';
    }
    else {

        echo '{"error": "teste"}';
    }
?>
```

Note que o método echo é responsável por retornar os dados como resposta para a requisição.

Atividades:

1. Criar um formulário com os seguintes campos:

- Nome (input text)
- Telefone (input tel)
- Cidade (input text)
- Estado (select)
- Data de Nascimento (input date)
- Usuário (input text)
- Password (input password)
- Input Submit e Input Reset

Obs.: utilizar method = GET

O formulário deve enviar os dados para um código PHP (Arquivo anexo). Altere o method de GET para POST e altere a forma de acesso aos dados no PHP de $_GET[] para $_POST[].

2. Altere o código do exercício 1 para utilizar o JavaScript para enviar as requisições para o servidor.