Exercícios – Aula 21

Arquivos - Exercícios

1) Prepare um formulário html (up.html) para enviar arquivos. Neste exemplo enviaremos um documento pdf e uma imagem. Observe atentamente o código abaixo:

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Envio de arquivos</title>
7  </head>
8  <body>
9      <h3>Envio de arquivos via HTML</h3>
10
11     <form action="upload1.php" method="POST" enctype="multipart/form-data">
12         <label for="autor">Autor do envio: </label>
13         <input type="text" name="autor" id="autor"><br>
14         <label for="arquivo1">Documento: </label>
15         <input type="file" name="arquivo1" id="arquivo1" accept="application/pdf"><br>
16         <label for="arquivo2">Imagem: </label>
17         <input type="file" name="arquivo2" id="arquivo2" accept="image/*"><br>
18         <input type="submit" value="enviar">
19     </form>
20 </body>
21 </html>
```

Considerações importantes:

Linha 11 → O enctype (formato de envio) precisa ser multipart/form-data.
Linha 15 → Em accept definimos que só é possível enviar um arquivo pdf (veja opções consultando a tabela de MIME/TYPES na apostila).
Linha 17 → Em accept definimos que só é possível enviar um arquivo de imagem. No entanto, não especificamos a extensão.

2) Dentro do seu projeto, crie a pasta up para receber os arquivos.
3) Escreva o arquivo upload1.php que mostrará as propriedades de cada arquivo através da variável global $_FILES e o conteúdo enviado pelo método POST que nesse caso é o autor do envio. Ao final os arquivos enviados serão movidos para a pasta up.

upload1.php > ...

```
1  <?php
2      echo "<pre>";
3      print_r($_FILES);
4      echo "</pre>";
5      echo "<hr>";
6      echo "<pre>";
7      print_r($_POST);
8      echo "</pre>";
9      $dirUpload = "up/";
10     foreach($_FILES as $arquivo){
11         $arquivoOrigem = $arquivo['tmp_name'];
12         $arquivoDestino = $dirUpload.$arquivo['name'];
13         if(move_uploaded_file($arquivoOrigem,$arquivoDestino))
14             echo "Arquivo salvo com o caminho {$arquivoDestino}<br>";
15         else
16             echo "Erro ao fazer upload de arquivo<br>";
17     }
```

Importante:
Linha 9 → Declaramos o diretório padrão de upload.
Linhas 10 a 17 → Percorremos os arquivos enviados movendo-os para o diretório de upload por meio da função nativa move_uploaded_files.

4) Veja como deverá ficar no navegador.

```
Array
(
    [arquivo1] => Array
        (
            [name] => Ata2023.06.26.pdf
            [full_path] => Ata2023.06.26.pdf
            [type] => application/pdf
            [tmp_name] => C:\xampp\tmp\php2950.tmp
            [error] => 0
            [size] => 404810
        )

    [arquivo2] => Array
        (
            [name] => eu.jpg
            [full_path] => eu.jpg
            [type] => image/jpeg
            [tmp_name] => C:\xampp\tmp\php2961.tmp
            [error] => 0
            [size] => 13480
        )

)
```

```
Array
(
    [autor] => Rafael
)
```

Arquivo salvo com o caminho up/Ata2023.06.26.pdf
Arquivo salvo com o caminho up/eu.jpg

Arquivos – Exercícios parte 2 (Enviando múltiplos arquivos com [] e com a diretiva multiple)

1) Prepare um formulário html (up2.html) para enviar arquivos múltiplos. Neste exemplo enviaremos vários documentos pdf e uma imagem. No entanto, todos os arquivos serão enviados pelo name aqrquivos. Observe atentamente o código abaixo:

up2.html > html

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Envio de arquivos múltiplos</title>
7  </head>
8  <body>
9      <h3>Envio de arquivos via HTML (arquivos múltiplos)</h3>
10
11     <form action="upload2.php" method="POST" enctype="multipart/form-data">
12         <label for="autor">Autor do envio: </label>
13         <input type="text" name="autor" id="autor"><br>
14         <label for="arquivo1">Documento: </label>
15         <input type="file" name="arquivos[]" id="arquivo1" multiple accept="application/pdf"><br>
16         <label for="arquivo2">Imagem: </label>
17         <input type="file" name="arquivos[]" id="arquivo2" accept="image/*"><br>
18         <input type="submit" value="enviar">
19     </form>
20 </body>
21 </html>
```

Linha 11 → O enctype (formato de envio) precisa ser multipart/form-data.
Linha 15 → Antes de accept definimos que é possível enviar vários arquivos pdf e utilizando colchetes ([]) definimos que os dois inputs enviarão arquivos com o mesmo nome (arquivos).
Linha 17 → Em accept definimos que só é possível enviar um arquivo de imagem. No entanto, não especificamos a extensão. Esse input também envia o arquivo de imagem pelo nome arquivos já que faz uso dos colchetes para um mesmo name.

2) Escreva o arquivo upload2.php que mostrará as propriedades de cada arquivo (todos sob o name arquivos) através da variável global $_FILES e o conteúdo enviado pelo

método POST que nesse caso é o autor do envio. Ao final os arquivos enviados serão movidos para a pasta up. Observe atentamente as diferenças. Se não entender algo, pergunte ao seu professor.

upload2.php > ...

```php
<?php
    echo "<pre>";
    print_r($_FILES);
    echo "</pre>";
    echo "<hr>";
    echo "<pre>";
    print_r($_POST);
    echo "</pre>";
    echo $_POST["autor"]."<br><hr>";
    $dirUpload = "up/";
    $nomes = $_FILES['arquivos']['name'];
    $nomesTmp = $_FILES['arquivos']['tmp_name'];
    for($i=0;$i<count($nomes);$i++){
            $arqOrigem = $nomesTmp[$i];
            $arqDestino = $dirUpload.$nomes[$i];
            if(move_uploaded_file($arqOrigem,$arqDestino))
                echo "Upload be sucedido. Caminho: {$arqDestino}<br>";
            else
                echo "Erro ao fazer upload de arquivo {$nomes[$i]}<br>";
    }
```

Importante:
Linha 11 → $nomes está recebendo um array com a propriedade name de todos os arquivos enviados.
Linha 12 → $nomesTmp está recebendo um array com a propriedade tmp_name de todos os arquivos enviados.
Linhas 13 a 20 → A cada iteração (a cada arquivo) $arqOrigem recebe o nome temporário do arquivo a ser movido e $arqDestino recebe o nome do arquivo destino precedido pelo diretório de upload. A função move_uploaded_files cuida da movimentação dos arquivos a cada iteração.
Em caso de dúvidas, consulte o seu professor.

3) Veja como deverá ficar no navegador.

```
Array
(
    [arquivos] => Array
        (
            [name] => Array
                (
                    [0] => Exame outubro (1).pdf
                    [1] => 272910458 EXAME AGOSTO (1).pdf
                    [2] => 272910458 EXAME AGOSTO.pdf
                    [3] => eu.jpg
                )

            [full_path] => Array
                (
                    [0] => Exame outubro (1).pdf
                    [1] => 272910458 EXAME AGOSTO (1).pdf
                    [2] => 272910458 EXAME AGOSTO.pdf
                    [3] => eu.jpg
                )

            [type] => Array
                (
                    [0] => application/pdf
                    [1] => application/pdf
                    [2] => application/pdf
                    [3] => image/jpeg
                )

            [tmp_name] => Array
                (
                    [0] => C:\xampp\tmp\php952B.tmp
                    [1] => C:\xampp\tmp\php957A.tmp
                    [2] => C:\xampp\tmp\php95AA.tmp
                    [3] => C:\xampp\tmp\php95CA.tmp
                )

            [error] => Array
                (
                    [0] => 0
                    [1] => 0
                    [2] => 0
                    [3] => 0
                )

            [size] => Array
                (
                    [0] => 3454786
                    [1] => 825109
                    [2] => 825109
                    [3] => 13480
                )

        )

)
```

```
Array
(
    [autor] => Rafael
)
```

Rafael

Upload be sucedido. Caminho: up/Exame outubro (1).pdf
Upload be sucedido. Caminho: up/272910458 EXAME AGOSTO (1).pdf
Upload be sucedido. Caminho: up/272910458 EXAME AGOSTO.pdf
Upload be sucedido. Caminho: up/eu.jpg

Exercícios – Parte 3

1) Crie um arquivo PHP para testar o uso de array_map com um array unidimensional.

```php
$numeros = [1,2,3,4,5];
# array_map(<callback>, <array>):<array>
# a cada iteração minha função de callback tem um elemento float
$numeros2 = array_map(
    function(int $n){
        return $n*$n*$n;
    },
    $numeros);
#com arrow function
$numeros2 = array_map(fn(int $n)=>$n*$n*$n,$numeros);
#imprimindo a estrutura
echo "<pre>";
print_r($numeros2);
echo "</pre>";
```

2) Agora crie outro arquivo para entender como utilizar array_map para um array multidimensional.

```php
#Uma abordagem mais complexa
$alunos = [
    ["nome"=>"Rafael","notas"=>[3.5,6.5]],
    ["nome"=>"Renata","notas"=>[4.5,7.5]]
];
echo "<pre>";
print_r($alunos);
echo "</pre>";
#Trabalhando com um array multidimensional
#A cada iteração minha função de callback tem um elemento um array com duas informações
$alunos2 = array_map(
    function(array $linha){
        $media = array_sum($linha['notas'])/count($linha['notas']);
        # A cada iteração $alunos recebe um novo array agoira com uma 3ª informação
        return ["nome"=>$linha['nome'], "notas"=>$linha['notas'], "media"=>$media];
    },
    $alunos);
#imprimindo a estrutura
echo "<pre>";
print_r($alunos2);
echo "</pre>";
```

3) Agora crie outro exemplo para entender como utilizar array_filter.

```php
#array_filter
$numeros = [1,2,3,4,5];
$numerosPares = array_filter($numeros, function(int $n):int{return ($n%2)===0;});
$numerosPares = array_filter($numeros, fn(int $n):int=>($n%2)===0);
#imprimindo a estrutura
echo "<pre>";
print_r($numerosPares);
echo "</pre>";
```