PHP - Arquivos

Revisão

NA AULA PASSADA...

- Embed
- Sintaxe curta
  - if: - endif
  - foreach: - endforeach
- Métodos
  - GET e POST
- Validação de Formulários
- Redirect
- Debug

DigitalHouse > Coding School

Arquivos

O PHP permite gerenciar arquivos como você quiser, tanto para processamento quanto para suporte. Nosso foco serão os arquivos JSON.

FILE_EXISTS

A função **file_exists()** é utilizada para verificar se um arquivo existe.

Ela recebe um único parâmetro, o arquivo que desejamos verificar a existência Exemplo:

file_exists(string **$nomeDoArquivo**)

FILE_EXISTS - Exemplo

Iremos verificar se o arquivo **dados.txt** existe com o auxílio da função **file_exists().**

```
<?php
    $arquivo = 'dados.txt';
    if (file_exists($arquivo)) {
        echo "O arquivo $arquivo existe";
    } else {
        echo "O arquivo $arquivo não existe";
    }
?>
```

FOPEN

A função **fopen()** é utilizada para abrir um arquivo ou uma URL e **retorna um recurso** (resource).

Ela recebe dois parâmetros, onde o primeiro é o nome do arquivo e o segundo o modo de abertura do arquivo indicado. Exemplo:

```
fopen(string $nomeDoArquivo, string $modoDeAbertura)
```

FOPEN - Modos de Abertura de Arquivo

Alguns modos:

- **r :** Abre **somente para leitura**; coloca o ponteiro do arquivo no começo do arquivo.
- **w :** Abre **somente para escrita**; coloca o ponteiro do arquivo no começo do arquivo e reduz o comprimento do arquivo para zero. Se o arquivo não existir, tenta criá-lo.
- **a :** Abre **somente para escrita**; coloca o ponteiro do arquivo no final do arquivo. Se o arquivo não existir, tenta criá-lo.
- **x :** Cria e abre o arquivo **somente para escrita**; coloca o ponteiro no começo do arquivo. Se o arquivo já existir, a chamada a fopen() falhará, retornando FALSE e gerando um erro de nível E_WARNING. Se o arquivo não existir, tenta criá-lo.

FOPEN - Modos de Abertura de Arquivo

Alguns modos:

- **r+ :** Abre para **leitura e escrita**; coloca o ponteiro do arquivo no começo do arquivo.
- **w+ :** Abre para **leitura e escrita**; coloca o ponteiro do arquivo no começo do arquivo e reduz o comprimento do arquivo para zero. Se o arquivo não existir, tenta criá-lo.
- **a+ :** Abre para **leitura e escrita**; coloca o ponteiro do arquivo no final do arquivo. Se o arquivo não existir, tenta criá-lo.
- **x+ :** Cria e abre o arquivo para **leitura e escrita**; coloca o ponteiro no começo do arquivo. Se o arquivo já existir, a chamada a fopen() falhará, retornando FALSE e gerando um erro de nível E_WARNING. Se o arquivo não existir, tenta criá-lo.

FOPEN - Exemplo

Iremos criar uma variável **$arquivo** que irá receber a string **“texto.txt”**.
Portanto esta variável **$arquivo** será o primeiro parâmetro informado na função **fopen**.
O segundo parâmetro será a string **“r”**,
Isto irá indicar que o arquivo será aberto para leitura e posiciona o ponteiro do arquivo no início do mesmo.

```
<?php
    $arquivo = “texto.txt”;
    $abrindoArquivo = fopen($arquivo, “r”);
?>
```

FWRITE

A função **fwrite()** é utilizada para escrever um arquivo.

Ela recebe dois parâmetros, onde o primeiro é o recurso (arquivo com algum modo de abertura) e o segundo o texto que desejamos inserir neste arquivo indicado. Exemplo:

fwrite(resource **$recurso**, string **$texto**)

FWRITE - Exemplo

Observe que iremos primeiro utilizar a função **fopen()**, para abrirmos o arquivo **texto.txt** e somente após iremos utilizar a função **fwrite()** para escrever **123** no arquivo **texto.txt.**

```
<?php
    $arquivo = fopen('texto.txt', 'w');
    fwrite($arquivo, '1-2-3');
?>
```

FCLOSE

A função **fclose()** é utilizada para fechar um arquivo que foi aberto.

Ela recebe um único parâmetro, o recurso (qualquer arquivo que foi aberto com algum modo de abertura)

fclose(resource **$recurso**)

FCLOSE - Exemplo

Observe que abrimos o arquivo **texto.txt**, escrevemos **123** neste arquivo e finalizamos fechando este arquivo.

```
<?php
    $arquivo = fopen('texto.txt', 'w');
    fwrite($arquivo, '1,2,3');
    fclose($arquivo);
?>
```

Gerenciamento de Arquivos

```php
<?php
    // abrir um arquivo específico com fopen()
    $fp = fopen('dados.txt', 'w');
    // escrever algo dentro deste arquivo com fwrite()
    fwrite($fp, 'Olá alunos');
    // fechar o arquivo que foi aberto com fclose()
    fclose($fp);
?>
```

O conteúdo de '**dados.txt**' agora é **Olá alunos**

FILESIZE

A função **filesize()** é utilizada para obter o tamanho de um arquivo.

Ela recebe um único parâmetro, o nome do arquivo que desejamos obter o tamanho.

fwrite(string **$nomeDoArquivo**)

FILESIZE

Observe que abrimos o arquivo **texto.txt** com a função **fopen()**, e utilizamos a função **filesize()** para obter o tamanho do arquivo **texto.txt.**

```
<?php
    $arquivo = fopen('texto.txt', 'w');
    $tamanho = filesize('texto.txt');
?>
```

FREAD

A função **fread()** é utilizada para leitura um arquivo.

Ela recebe dois parâmetros, onde o primeiro é o recurso (arquivo com algum modo de abertura) e o segundo o tamanho do arquivo indicado. Exemplo:

fread(resource **$recurso**, int **$tamanho**)

FREAD

Observe que abrimos o arquivo "**texto.txt**" com a função **fopen()**, identificamos o tamanho dele através da função **filesize()**, e obtemos o conteúdo deste arquivo com a função **fread().**

```
<?php
    $arquivo = fopen('texto.txt', 'w');
    $tamanho = filesize('texto.txt');
    $conteudo = fread($arquivo, $tamanho);
?>
```

FGETS

A função **fgets()** é utilizada para retornar uma linha de um arquivo aberto.

Ela recebe um único parâmetro, o recurso (qualquer arquivo que foi aberto com algum modo de abertura). Exemplo:

fgets(resource **$recurso**)

FILE_GET_CONTENTS

A função **file_get_contents()** é utilizada para leitura de um arquivo inteiro em uma string.
Funciona como uma espécie de de atalho se comparado a sequência de comandos: **fopen** -> **fread** -> **fclose**

Ela recebe um único parâmetro, o nome do arquivo indicado.Exemplo:

file_get_contents(string **$nomeDoArquivo**)

FILE_GET_CONTENTS

Observe que iremos obter e ler o conteúdo do arquivo “**texto.txt**” com a função **file_get_contents()**.

```
<?php
    $arquivo = file_get_contens('texto.txt');
    echo $arquivo;
?>
```

FGETS - Exemplo

Observe que iremos obter todas as linhas do arquivo **dados.txt** utilizando um **while** para percorrer linha a linha deste arquivo.

```
<?php
    $arquivo = fopen("dados.txt", "r");
    if ($arquivo) {
        while (($linha = fgets($arquivo)) {
            echo $linha;
        }
    }
    fclose($arquivo);
?>
```

FILE_PUT_CONTENTS

A função **file_put_contents()** é utilizada para gravar dados em um arquivo.
Funciona como uma espécie de de atalho se comparado a sequência de comandos: **fopen** -> **fwrite** -> **fclose**

Ela recebe dois parâmetros, onde o primeiro é nome do arquivo indicado, e o segundo é o conteúdo a ser inserido. Exemplo:

file_put_contents(string **$nomeDoArquivo**, string **$texto**)

FILE_PUT_CONTENTS

Observe que iremos obter e ler o conteúdo do arquivo "**texto.txt"** com a função **file_get_contents()**. E em sequência adicionar o texto "**Full Stack**", com a função **file_put_contents().**

```
<?php
    $arquivo = file_get_contents('texto.txt');
    $arquivo .= "Full Stack\n";
    file_put_contents('texto.txt', $arquivo);
?>
```

É possível ler linha por linha?

- Ler arquivos de texto **muito grandes** pode provocar **problemas de memória**.

- Podemos utilizar **fread** ou **file_get_contents** e explodir o conteúdo com o caractere **\n**.

JSON

JSON, acrônimo de JavaScript Object Notation, é um formato de texto leve para a troca de dados, baseado em objeto de JavaScript.

JSON - Exemplo de Formato JSON

```
{
    "usuarios" : [
        { "nome" : "John", "sobrenome" : "Doe" },
        { "nome" : "Anna", "sobrenome" : "Smith"},
        { "nome" : "Peter", "sobrenome" : "Jones" }
    ]
}
```

Funções PHP JSON

O PHP oferece 2 funções para operar com variáveis com conteúdo correspondente a um JSON.

Onde **json_encode()**, é utilizada para gerar um JSON.

- **json_encode()**

Já a função **json_decode()**, transforma um JSON em um objeto contendo um array associativo.

- **json_decode()**

JSON_ENCODE

Observe que iremos criar um array associativo e depois transformar este array em um json.

```php
<?php
    $carro = [
        "Marca" => "Ford",
        "Cor" => "Preto"
    ];
    json_encode($carro);
?>
```

JSON Gerado

```
{
    "Marca": "Ford",
    "Cor": "Preto"
}
```

DigitalHouse > Coding School

JSON_DECODE

Observe que agora iremos transformar um json em um **objeto** contendo um array associativo.

```
<?php
    $carro = [
        "Marca" => "Ford",
        "Cor" => "Preto"
    ];
    $json = json_encode($carro);
    json_decode($json);
?>
```

Objeto Gerado

```
stdClass Object {
   [Marca] => "Ford"
   [Cor] => "Preto"
}
```

DigitalHouse > Coding School

JSON_DECODE

Observe que agora iremos transformar um json em um **array associativo**, Para isso utilizaremos **true** no json_encode

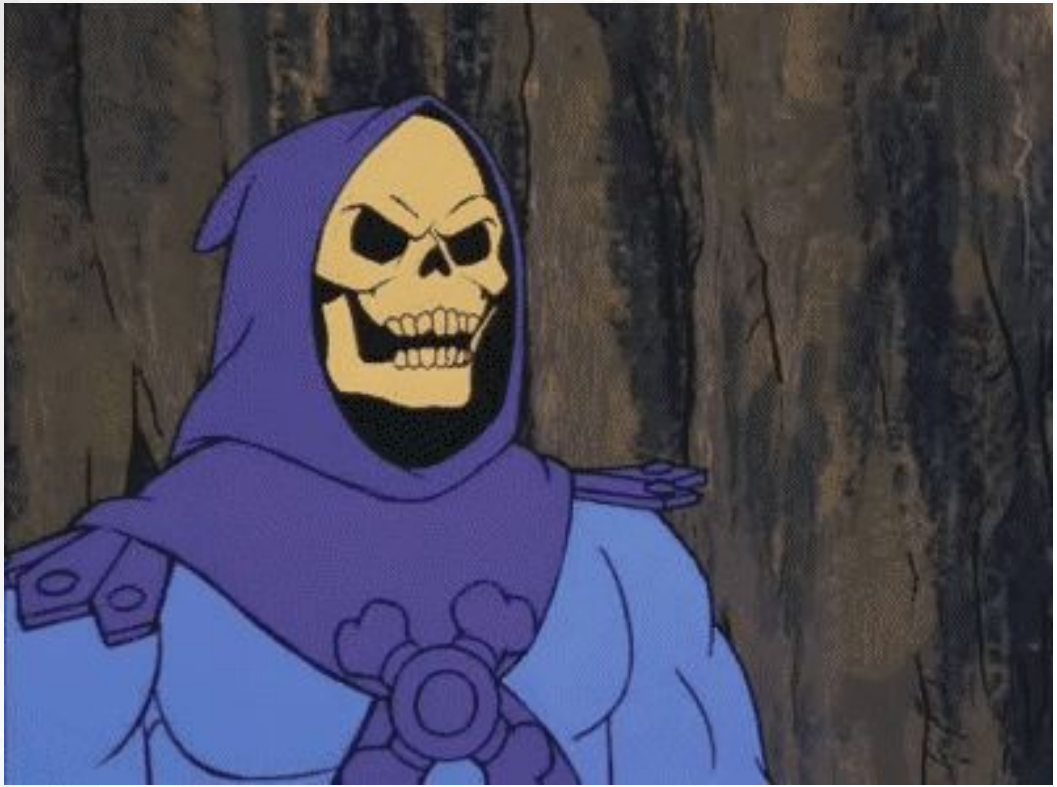
```
<?php
    $carro = [
        "Marca" => "Ford",
        "Cor" => "Preto"
    ];
    $json = json_encode($carro);
    json_decode($json, true);
?>
```

Array Associativo Gerado

```
array(2) {
   [Marca] => "Ford"
   [Cor] => "Preto"
}
```

DigitalHouse > Coding School

Obrigado até a próxima



DigitalHouse >
Coding School