DigitalHouse >
Coding School

Hashing e Upload

FULLSTACK - CAMPUS VILA OLÍMPIA
TURMA 03 - JULHO/2018

Aula passada...

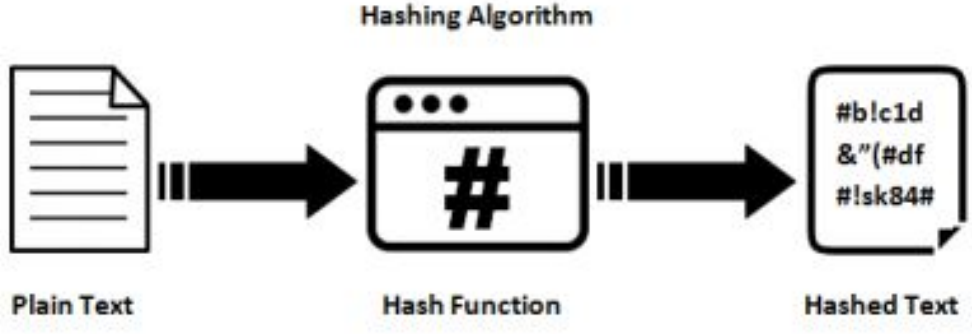
- JSON
- json_encode
- json_decode
- Manipulação de Arquivos
- fopen, fwrite, fread, fclose
- file_get_contents
- file_put_contentes
- fgets

Como eu faço para guardar senhas de forma segura?

CRIPTOGRAFIA!

HASHING

Uma função de hash é uma função **unilateral** que pega qualquer texto e transforma em um código criptografado que não pode voltar atrás



HASHING

Existem diferentes algoritmos de hashing, e o PHP permite utilizá-los tanto para criptografar quanto para verificar as informações

Ele é muito usado para a implementação de senhas para poder armazenar de forma segura.

MD5 string alfa-numérica de 32 caracteres

SHA1 string alfa-numérica de 40 caracteres alfa-numéricos.

BCRYPT string alfa-numérica de 60 a 255 caracteres alfa-numéricos.

password_hash

Cria um hash de um texto
(normalmente uma senha)

password_hash(*string* **$password**, *int* **$algoritmo**);

Cada vez que é executado a função retorna um hash diferente.

password_hash

```
<?php
      $hash = password_hash("senha123 ", PASSWORD_DEFAULT);
?>
```

A função password_hash retorna uma string com um hash, exemplo:

$2y$10$JA0PAm23yzblu7xOlHyz3uMJHddAQOW95W6ezTf7wcgC60GmtkEKK

Como faço pra verificar a senha?

password_verify(“senha123”, $hash);

Essa função compara a senha que o usuário digitou e senha salva no cadastro.

password_verify

```php
<?php

    $hash = password_hash("senha123", PASSWORD_DEFAULT);

    password_verify("senha123", $hash); //true
    password_verify("teste123", $hash); //false

?>
```

Vamos praticar!

Exercícios 1 e 2

Deixando o formulário mais maneiro:

Como faço para receber arquivos do usuário?

UPLOADS

O PHP oferece certas facilidades para gerenciar a carga de arquivos em formulários:

Formulário

```
<form action="inscrever.php" method="post">
    Nome: <input type="text" name="nome">
    Sobrenome: <input type="text" name="sobrenome">
    <input type="submit" value="Enviar">
</form>
```

Formulário

```
<form action="inscrever.php" method="post">
    Nome: <input type="text" name="nome">
    Sobrenome: <input type="text" name="sobrenome">
    Avatar: <input type="file" name="imgPerfil">
    <input type="submit" value="Enviar">
</form>
```

Formulário

```
<form action="inscrever.php" method="post"
      enctype="multipart/form-data">
    Nome: <input type="text" name="nome">
    Sobrenome: <input type="text" name="sobrenome">
    Avatar: <input type="file" name="imgPerfil">
    <input type="submit" value="Enviar">
</form>
```

form enctype

- **application/x-www-form-urlencoded** - Padrão. Todos os caracteres são codificados para o formato de URL antes do envio (espaços são convertidos em "+", e caracteres especiais são convertidos em valores HEXADECIMAIS ASCII, ex: %5F).

- **multipart/form-data** - Nenhum caractere é codificado. Esse valor é necessário para formulários com upload de arquivos.

- **text/plain** - Espaços são convertidos em "+", mas nenhum caractere especial é codificado.

$_FILES

É um array associativo de elementos do upload feito através do método POST.

Utilizando o formulário mostrado que tem um input de type="**file**" e com name="**imgPerfil**", como pegamos o arquivo enviado?

$_FILES

Para cada arquivo, haverá um array associativo.

Neste exemplo, **$_FILES["imgPerfil"]** é um **array associativo** que contém pelo menos estas chaves interessantes:

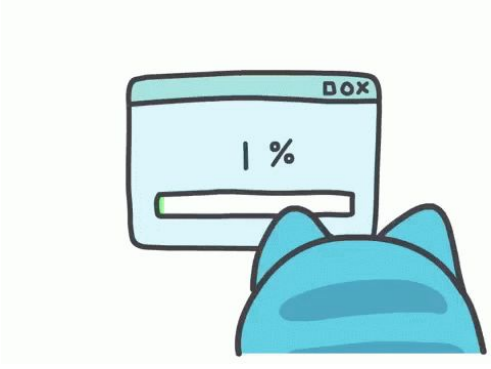
error - $_FILES["imgPerfil"]["error"]
name - $_FILES["imgPerfil"]["name"]
tmp_name - $_FILES["imgPerfil"]["tmp_name"]

$_FILES - error

Indica se o upload da imagem foi concluído.

Normalmente, é comparada com a constante **UPLOAD_ERR_OK** que indica que não houve erros.

```
if($_FILES["imgPerfil"]["error"] == UPLOAD_ERR_OK)
```



$_FILES - name

Name indica o nome do arquivo carregado.

```
<?php
    $nome = $_FILES["imgPerfil"]["name"];
?>
```

A variável **$nome** terá o nome original do arquivo.

$_FILES - tmp_name

É o endereço do arquivo **enviado** que foi gravado em uma **pasta temporária**.

```
<?php
    $nome=$_FILES["imgPerfil"]["name"];
    $arquivo=$_FILES["imgPerfil"]["tmp_name"];
?>
```

$_FILES - move_uploaded_file

Para salvar o arquivo em uma pasta, na verdade movemos ele da **pasta temporária** para **o lugar que queremos** com **move_uploaded_file**.

```
<?php
    $nome = $_FILES["imgPerfil"]["name"];
    $arquivo = $_FILES["imgPerfil"]["tmp_name"];
    $ok = move_uploaded_file($arquivo, $nome);
?>
```

$_FILES - move_uploaded_file

Caso queira gravar em uma pasta chamada “uploads”, no mesmo diretório do nosso php:

```
<?php
    $nome = $_FILES["imgPerfil"]["name"];
    $arquivo = $_FILES["imgPerfil"]["tmp_name"];
    $caminho = "uploads/".$nome;
    $ok = move_uploaded_file($arquivo, $caminho);
?>
```

dirname & __FILE__

Retorna o caminho completo do arquivo php que chamou a função:

```
<?php
    echo __FILE__;
?>
```

Retorna o caminho completo da pasta em que o arquivo se encontra:

```
<?php
    $diretorio = dirname(__FILE__);
?>
```

opendir & readdir

Servem para sabermos os nomes dos arquivos que existem em uma pasta ou diretório:

```
<?php
    if ($handle = opendir('.')) {
        while (false !== ($file = readdir($handle))) {
            if ($file != "." && $file != "..") {
                echo "$file\n";
            }
        }
        closedir($handle);
    }
?>
```

Vamos praticar!

Exercício 3

DAILY SCRUM

maximo 15min

- **o que fiz ontem?**
- **o que estou fazendo?**
- **tem algum impedimento?**

dica: use um cronometro!

OBRIGADO!

#SEXTOU