



PHP

Clase 7

Persistencia de datos

¿Qué pasa cuando enviamos un formulario y nos devuelve un error de validación?

Tu Nombre:

Tu Email:

Asunto:

Ingrese un email correcto



Persistencia de datos

Podemos utilizar **\$_POST** y así modificar el atributo value del input.



```
<input type="text" name="firstname" value="">
```

¡A practicar!

Ejercicio 1



```
<?php  
    echo "Hora de practicar!";  
?>
```

JSON

JavaScript Object Notation



Es un formato de texto ligero para el intercambio de datos. Leerlo y escribirlo es simple para humanos, mientras que para las máquinas es simple interpretarlo y generarlo.





JSON - Ejemplo

```
{  
  "employees": [  
    { "firstName": "John", "lastName": "Doe" },  
    { "firstName": "Anna", "lastName": "Smith" },  
    { "firstName": "Peter", "lastName": "Jones" }  
  ]  
}
```

JSON

JavaScript Object Notation



PHP nos proporciona 2 funciones para operar con variables cuyo contenido se corresponda con un JSON.

- `json_encode`
- `json_decode`



JSON

JavaScript Object Notation

json_encode

Retorna la representación JSON del valor dado

```
<?php
$auto = [
    "Marca" => "Ford",
    "Color" => "Negro"
];
json_encode($auto);
?>
```


JSON

JavaScript Object Notation

json_encode

Decodifica un string de JSON

```
<?php
    $auto = [
        "Marca"=> "Ford",
        "Color"=> "Negro"
    ];
    $json = json_encode($auto);

    json_decode($json);
?>
```

¡Devuelve un objeto!

JSON

JavaScript Object Notation

json_decode

Decodifica un string de JSON

```
<?php
    $auto = [
        "Marca"=> "Ford",
        "Color"=> "Negro"
    ];
    $json = json_encode($auto);

    json_decode($json, true);
?>
```

Devuelve un array asociativo

¡A practicar!

Persistencia de datos

Ejercicio 1

JSON

Ejercicio 1



```
<?php  
    echo "Hora de practicar!";  
?>
```

PHP

Archivos



PHP nos permite manejar archivos a gusto, tanto para procesamiento como para **soporte**.

Nosotros nos enfocaremos en archivos **JSON**.





Manejo de archivos

```
<?php
```

```
    $fp = fopen('data.txt', 'w');  
    fwrite($fp, '1');  
    fwrite($fp, '23');  
    fclose($fp);
```

```
?>
```

¡El contenido de 'data.txt' ahora es 123.




fopen

Abre un fichero o un URL y **retorna un recurso**.

fopen(*string* **\$filename**, *string* **\$mode**)

Algunos modos:

- r
 - r+
 - w
 - w+
 - a
 - a+
- 



fwrite

Escribe un archivo.

`fwrite(recurso $recurso, string $texto)`



¡No es lo mismo hacer **fopen** en modo **r** que en modo **a**!





fclose

Cierra un archivo abierto.

`fclose(recurso $recurso)`



fread

Lectura de un archivo.

`fread(recurso $recurso, int $length)`

```
<?php
    $nombre_fichero = "/usr/local/algo.txt";
    $gestor = fopen($nombre_fichero, "r");
    $tamanio = filesize($nombre_fichero);
    $contenido = fread($gestor, $tamanio);
    fclose($gestor);
?>
```



file_get_contents

Levanta un archivo en una cadena de texto.

```
<?php
    $paginalInicio = file_get_contents("data.txt");
    echo $paginalInicio;
?>
```

Es una especie de atajo a utilizar:
fopen -> fread -> fclose

file_put_contents

Guarda una cadena de texto a un archivo.

```
<?php
    $archivo = "data.txt";
    $actual = file_get_contents($archivo);
    $actual .= "Juan Perez\n";
    file_put_contents($archivo, $actual);
?>
```

Es una especie de atajo a utilizar
fopen -> fwrite -> fclose

¡Cuidado, usado así, pisa todo el contenido del archivo!

file_put_contents

```
<?php
    $archivo = "data.txt";
    $persona = "Juan Perez\n";
    file_put_contents($archivo, $persona,
FILE_APPEND | LOCK_EX);
?>
```

¡Estos métodos nos permiten agregar o sobrescribir datos en archivos fácilmente!



¿Y como leemos línea por línea?

- Un archivo de texto puede ser muy grande y leerlo completo puede provocar problemas de memoria.
- Podemos utilizar **fread** o **file_get_contents** y explotar el contenido con el caracter `\n`.
- No parece la mejor idea... ¿Entonces?

fgets

Devuelve una línea de un archivo abierto

```
<?php
    $gestor = fopen("data.txt", "r");
    if ($gestor) {
        while (($linea = fgets($gestor)) !== false) {
            echo $linea;
        }
    }
    fclose($gestor);
?>
```

file_exists

Comprueba si un archivo existe

```
<?php
    $archivo = 'data.txt';
    if (file_exists($archivo)) {
        echo "El archivo $archivo existe";
    } else {
        echo "El archivo $archivo NO existe";
    }
?>
```

PHP

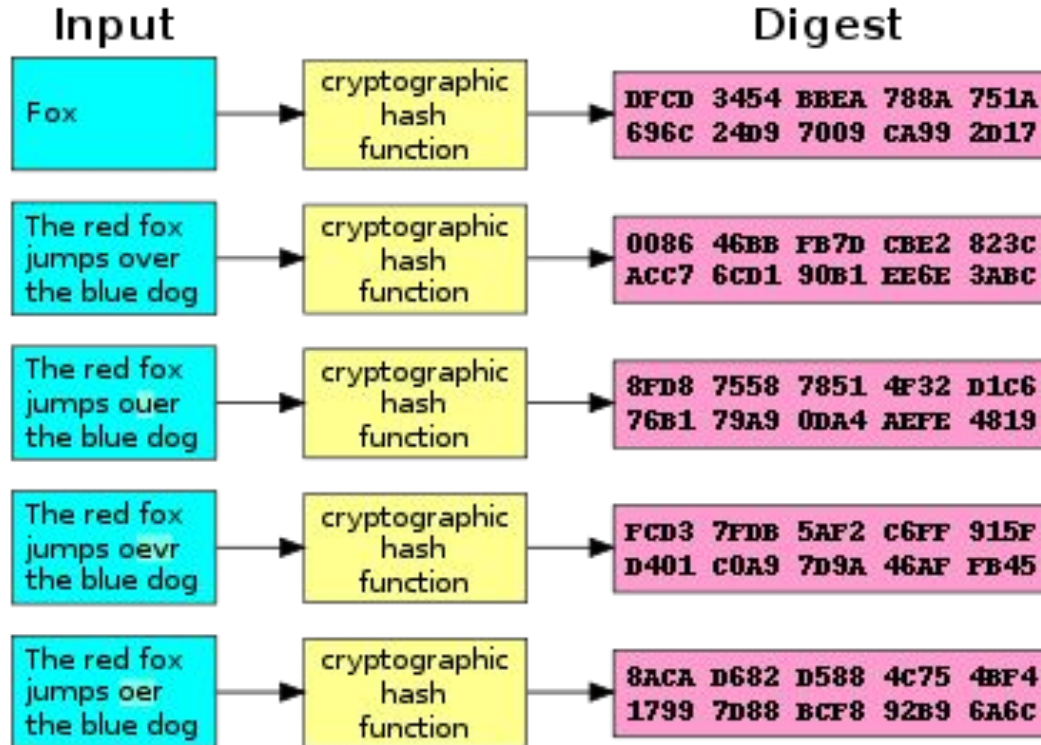
Hashing



Una función de hash es una función **unilateral** que toma cualquier texto y lo transforma en un código encriptado que no se puede volver atrás.



Hashing



PHP

Hashing



Existen diversos algoritmos de hasheo y PHP nos permite utilizarlas tanto para encriptar como verificar información.

Gran parte de su uso se ve en la implementación de contraseñas para poder almacenarlas de forma segura.



password_hash

Crea un hash de un texto (habitualmente una contraseña).

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

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

password_verify

Comprueba que un texto (habitualmente una contraseña) coincida con un hash.

```
password_verify(string $password, string $hash);
```

```
<?php
```

```
    $hash1 = password_hash("Hola", PASSWORD_DEFAULT);  
    $hash2 = password_hash("Chau", PASSWORD_DEFAULT);
```

```
    password_verify("Hola", $hash1);  
    password_verify("Chau", $hash2);
```

```
?>
```

¡A practicar!

Ejercicios del **3** al **7**



```
<?php  
    echo "Hora de practicar!";  
?>
```

¡Gracias!



¿Preguntas?