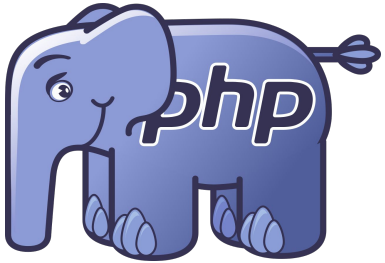
Information Technology Institute



Day02

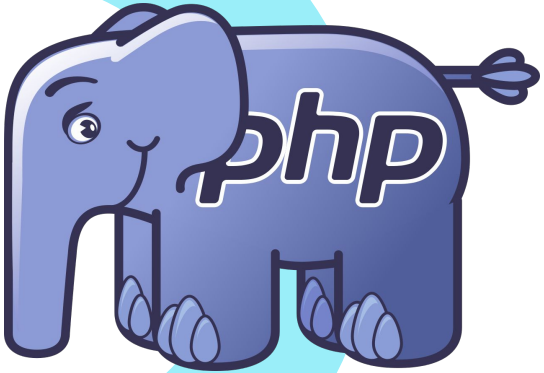
Prepared by:

Noha Shehab
Teaching Assistant
Information Technology Institute (ITI)

Agenda

- Files
  - Open file
  - Read from file
  - Write to file
  - Close the file
- Arrays
  - Simple arrays
  - Associative arrays

Files in



Storing data

- You can store data in two basic ways: in flat files or in a database.

- A flat file can have many formats, but in general, when we refer to a flat file, we mean a simple **text file**.

Processing files(Reading)

- Writing data to a file requires three steps:
  - Open the file. If you cannot open the file (for example, if it doesn't exist), you need to recognize this and exit gracefully.
  - Read the data from the file.
  - Close the file.

Processing files(Writing)

- Writing data to a file requires three steps:
  - Open the file. If the file doesn't already exist, you need to create it.
  - Write the data to the file.
  - Close the file.

Opening the file

- To open a file in PHP, you use the **fopen()** function.
- When you open the file, you need to specify how you intend to use it. This is known as the file mode.

```
fopen( string $filename , string $mode)    Resource
```

File modes

You need to make three choices when opening a file:

- You might want to open a file for reading only, for writing only, or for both reading and writing.
- If writing to a file, you might want to overwrite any existing contents of a file or append new data to the end of the file. You also might like to terminate your program gracefully instead of overwriting a file if the file already exists.

The **fopen()** function supports combinations of these three options.

File modes

Mode	Mode Name	Result
r	Read	Open the file for reading, beginning from the start of the file.
r+	Read	Open the file for reading and writing, beginning from the start of the file.
w	Write	Open the file for writing, beginning from the start of the file. If the file already exists, delete the existing contents. If it does not exist, try to create it.
w+	Write	Open the file for writing and reading, beginning from the start of the file. If the file already exists, delete the existing contents. If it does not exist, try to create it.
x	Cautious write	Open the file for writing, beginning from the start of the file. If the file already exists, it will not be opened, fopen() will return false, and PHP will generate a warning.

File modes

Mode	Mode Name	Result
x+	Cautious write	Open the file for writing and reading, beginning from the start of the file. If the file already exists, it will not be opened, fopen() will return false, and PHP will generate a warning.
a	Append	Open the file for appending (writing) only, starting from the end of the existing contents, if any. If it does not exist, try to create it.
a+	Append	Open the file for appending (writing) and reading, starting from the end of the existing contents, if any. If it does not exist, try to create it.
b	Binary	Used in conjunction with one of the other modes. You might want to use this mode if your file system differentiates between binary and text files. Windows systems differentiate; Unix systems do not. The PHP developers recommend you always use this option for maxi-mum portability. It is the default mode.
t	Text	Used in conjunction with one of the other modes. This mode is an option only in Windows systems. It is not recommended except before you have ported your code to work with the b option.

Open file on read mode

```php
$filehandler=fopen("welcome.txt","r");
var_dump($filehandler);
```

```
C:\wamp64\www\PHPSmart\Day02\filehandling.php:5:resource(3, stream)
```

Fopen will return with a resource object if the file found, if not it will return with false. And gives a warning,
You can you the @ error suppressor to display your own error message

```php
@ $file2=fopen("abc.txt","r");
if($file2){
    echo "File exists";
}else{
    echo "File not found";
}
```

Writing to a file

- You can use either of the functions **fwrite()** (file write) or **fputs**() fputs() is an alias to fwrite().
- The function fwrite() actually takes three parameters, but the third one is optional.
- The prototype for fwrite() is :
  int fwrite (resource handle, string string [, int length])
- The third parameter, length, is the maximum number of bytes to write.
- Close the file

```
$filehandler=fopen("welcome.txt","w");
fwrite($filehandler,"I am writing using fwrite");
fclose( $filehandler);
```

Reading from a file

- You open the file by using fopen()
- Check the file size
- Open file using fread()
- Close file

```php
$filehandler=fopen("welcome.txt","r");
$filesize=filesize("welcome.txt");
$data=fread($filehandler,$filesize);
var_dump($data);
fclose( $filehandler);
```

```
C:\wamp64\www\PHPSmart\Day02\filehandling.php:24:string 'I am writing using fwrite' (length=25)
```

Reading from a file functions

- Foef():**file end of file** function takes a file handle as its single parameter. It returns true if the file pointer is at the end of the file.
- Fgets():This function **reads one line at a time** from a file
- Fgets(): can accept length n, and returned with the specified length with n-1

```php
$filehandler=fopen("welcome.txt","r");
$filesize=filesize("welcome.txt");
while (!feof($filehandler)){
    echo fgets($filehandler). "<br>";
}
fclose( $filehandler);
```

I am writing using fwrite
line 2
line 3
line 4
line 5

Reading from a file functions

- fgetcsv(): This function breaks **up lines of files** when you have used a **delimiting character**, such as the tab character or a comma.
- Reads the file into an array according the specified delimeter.
- **fgetcsv(file_pointer, length, delimiter, enclosure, escape)**

```
$filehandler=fopen("welcome.txt","r");
$filesize=filesize("welcome.txt");
    while (!feof($filehandler)){
        var_dump(fgetcsv($filehandler,100,"n")); ;
    }
    fclose( $filehandler);
```

Reading from a file functions

```php
$filehandler=fopen("welcome.txt","r");
$filesize=filesize("welcome.txt");
    while (!feof($filehandler)){
        var_dump(fgetcsv($filehandler,100,"n")); ;
    }
    fclose( $filehandler);
```

```
C:\wamp64\www\PHPSmart\Day02\filehandling.php:42:
array (size=1)
  0 => string 'I am writing using fwrite' (length=25)

C:\wamp64\www\PHPSmart\Day02\filehandling.php:44:
array (size=2)
  0 => string 'li' (length=2)
  1 => string 'e 2' (length=3)
```

Reading file in one step..

- Readfile: will return with the file content as a string without fopen or fclose

```
readfile("welcome.txt");
```

I am writing using fwrite line 2 line 3 line 4 line 5

- File($filename): will read the file and returns with its content in form of array.

```
$data= file("welcome.txt");
var_dump($data);
```

```
array (size=5)
  0 => string 'I am writing using fwrite
' (length=27)
  1 => string 'line 2
' (length=8)
  2 => string 'line 3
' (length=8)
  3 => string 'line 4
' (length=8)
  4 => string 'line 5' (length=6)
```

Reading file in one step..

- Fgetcontent(): This function is identical to readfile() except that it returns the content of the file as a string instead of outputting it to the browser.

```
$data= file_get_contents("welcome.txt");
var_dump($data);
```

```
C:\wamp64\www\PHPSmart\Day02\filehandling.php:63:string 'I am writing using fwrite
line 2
line 3
line 4
line 5' (length=57)
```

File useful functions.

rewind()

- function resets the file pointer to the beginning of the file.

ftell ()

- function reports how far into the file the pointe is in bytes.

fseek ()

- to set the file pointer to some point within the file.

Filetype()

- String filetype(string $filepath)
- Check type of file (file, folder)

file_exists():

- Checking Whether a File Is there.

unlink (file)

- Deleting a File

File useful functions.

Copy file: **copy(string $source, string $destination)**

is_dir(), is_executable() ,is_file(), is_link(), is_readable(), is_writable()

basename() function returns the filename from a path.

```
$path = "C:\wamp64\www\PHPSmart\Day02\\filehandling.php";
echo basename($path);
```

```
var_dump(is_executable($path));   // false
```

Locking files

- You need to pass it a pointer to an open file and a constant representing the kind of lock you require. It returns true if the lock was successfully acquired and false if it was not.

Value of Operation	Meaning
LOCK_SH	Reading lock. The file can be shared with other readers.
LOCK_EX	Writing lock. This operation is exclusive; the file cannot be shared
LOCK_UN	The existing lock is released.

Locking files

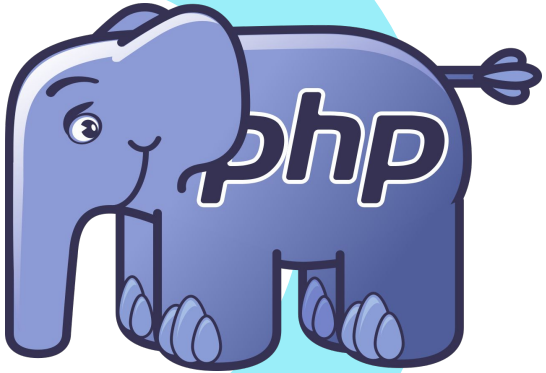
- To lock a file while you are writing to it. Use flock() function.
- This function should be called after a file has been opened but before any data is read from or written to the file.

```php
// flock($resource, locktype)
if (flock($file,LOCK_EX)) {
    fwrite($file,"Add some text to the file.");
    // release lock
    flock($file,LOCK_UN);
} else {
    echo "Error locking file!";
}
fclose($file);
```

File problems

- There are a number of problems in working with flat files:
- When a file grows large, working with it can be very slow.
- Searching for a particular record or group of records in a flat file
- is difficult. If the records are in order.
- Beyond the limits offered by file permissions, there is no easy way of enforcing different levels of access to data.
- Dealing with concurrent access can become problematic. You can lock files, but it can also cause a bottleneck.
- All of these file processing operations are sequential processing; you start from the beginning of the file and read through to the end.

Arrays



Arrays in PHP

- Arrays in PHP are
  - Indexed Arrays
    - The indices of numerically indexed arrays in PHP, start at zero by default, although you can alter this value.
  - Associative arrays
    - Instead of indices, Associative arrays are key indexed arrays, (Key, Value) pairs

Indexed Arrays

- Indexed arrays definition.

```php
$arr=[3,5,"Application",True,"PHP"];
$arr1= array("Noha","Engineering","ITI");
```

```
array (size=5)
  0 => int 3
  1 => int 5
  2 => string 'Application' (length=11)
  3 => boolean true
  4 => string 'PHP' (length=3)
```

- Accessing array elements via index

```php
for($i=0;$i< count($arr); $i++ ){
    echo $arr[$i]." ";
}
```

3 , 5 , Application , 1 , PHP

- Range function creates an ascending sequence of numbers stored in an array.

```php
$arange=range( start: 0, end: 10, step: 2);
```

```
array (size=6)
  0 => int 0
  1 => int 2
  2 => int 4
  3 => int 6
  4 => int 8
  5 => int 10
```

Looping array contents

- You can either use usual for loop or foreach

```php
$alphRange=range("A","Z", 4);
foreach ($alphRange as $char){
    echo $char." , ";
}
```

```
A , E , I , M , Q , U , Y
```

Associative Arrays

- Associative arrays are key indexed arrays, The following code creates an array with alphabets names as keys and orders as values.

```php
$info=["Name"=>"Noha","Email"=>"nshehab@iti.gov.eg","Track"=>"Application"];
foreach ($info as $key=>$value){
    echo $key." : ". $value."<br>";
}
```

```
Name : Noha
Email : nshehab@iti.gov.eg
Track : Application
```

```php
echo $info["Name"]; //Noha
```

- Adding an element to the array

```php
$info["Intake"]=35;
```

Creating an array from variables

- Use compact function, let's see..

```php
$name="Noha Shehab";
$email="nshehab@iti.gov.eg";
$infoo=compact( var_name: "name", ...var_names: "email");
```

```
array (size=2)
  'name' => string 'Noha Shehab' (length=11)
  'email' => string 'nshehab@iti.gov.eg' (length=18)
```

Array operators

Operator	Name	Use	Result
+	Union	$a+$b	Returns an array containing everything in $a and $b
==	Equality	$a == $b	Returns true if $a and $b have the same key and pairs
===	Identity	$a === $b	Returns true if $a and $b have the key and value pairs the same order
!=	Inequality	$a and $b	Returns true if $a and $b are not equal
<>	Inequality	$a or $b	Returns true if $a and $b are not equal
!==	Non-identity	$a xor $b	Returns true if $a and $b are not identical

Array operators' example

- Check this ....

```php
$num=[2,4,6,8,10];
$alphas=["a","b","c","d"];
$arr3= $num+$alphas;
var_dump($arr3);
```

- See the output, Guess why?

```
array (size=5)
  0 => int 2
  1 => int 4
  2 => int 6
  3 => int 8
  4 => int 10
```

Multi-dimensional Arrays

- Each location in the array can hold another array. This way, you can create a two-dimensional array.
- Multi-dimensional array could be simple or associative or mixed.

```php
$students=array(
    1=>array("Ali","IOT"),
    2=>array("Mostafa","Cloud"),
    3=>["Noha","Application"]
  );
```

Array sorting

- Sort function -> accepts an array and sort it ascendingly.
- Sort function is case sensitive. All capital letters come before all lowercase letters. So A is less than Z, but Z is less than b.

```php
$names = array( 'noha', "Fatma", "Dina", "Andrew","Shimaa","suliman" );
sort($names); // returns with the are sorted ascending.
var_dump($names);
```

- Sort can accept additional parameter to specify the sorting style **SORT_REGULAR (the default)**, **SORT_NUMERIC**, or **SORT_STRING**

Sorting Associative arrays

- The function asort() orders the array according to the value of each element.

```php
$prices = array( "meat"=>100, "sugar"=>10, "tea"=>8 );
asort($prices);
var_dump($prices);
```

```
array (size=3)
  'tea' => int 8
  'sugar' => int 10
  'meat' => int 100
```

- You can sort the associative array by keys also.

```php
$info=["Name"=>"Noha","Email"=>"nshehab@iti.gov.eg","Track"=>"Application"];
ksort($info);
var_dump($info);
```

```
array (size=3)
  'Email' => string 'nshehab@iti.gov.eg' (length=18)
  'Name' => string 'Noha' (length=4)
  'Track' => string 'Application' (length=11)
```

Reverse sort..

- rsort() :
  - Sort simple array descending.

- arsort()
  - Sort associative array descending according to the value.

- Krsort()
  - Sort associative array descending according to the key.

User-defined sort

- User can provide the comparison function; you can write a comparison function that returns the opposite values.

```php
function cmp($a, $b)
{
    if ($a == $b) {
        return 0;
    }
    return ($a < $b) ? -1 : 1;
}
$a = array(3, 2, 5, 6, 1);

usort($a, "cmp");

foreach ($a as $key => $value) {
    echo "$key: $value"."<br>";
}
```

```
0: 1
1: 2
2: 3
3: 5
4: 6
```

REORDERING ARRAYS

- User can provide the comparison function; you can write a comparison function that returns the opposite values.

shuffle()

- randomly reorders the elements of your array.

array_reverse()

- gives you a copy of your array with all the elements in reverse order.

array_push ()

- to add one new element to the end of an array

array_pop()

- removes and returns one element from the end of an array.

Array flip

- Used with Associative arrays.
- Exchange the keys with values.

```php
$colors = array(
    'one' => 'red',
    'two' => 'blue',
    'three' => 'yellow');
var_dump(array_flip($colors));
```

```
array (size=3)
  'red' => string 'one' (length=3)
  'blue' => string 'two' (length=3)
  'yellow' => string 'three' (length=5)
```

Loading file content into an array...

- Load data into array using the function file().
- Each line in the file becomes one element of an array.

Name	Track	Company
Andrew	Application	ITI
Noha	Application	ITI
Fatma	Cloud	ITI

```php
$Staff=file("csvfile.csv");
var_dump($Staff);
echo "<table border='2'>";
echo "<tr>
    <th>Name</th>
    <th>Track</th>
    <th>Company</th>
    </tr>";
foreach ($Staff as $record){
    $data=explode(",",$record);
    foreach ($data as $val){
        echo "<td>". $val."</td>";
    }
    echo "</tr>";
}
echo "</table>";
```

Array navigation

- To check if the value exists use: in_array().

```php
$fruits = ['banana', 'apple'];
$found = in_array('banana', $fruits); // true
```

- Every array has an internal pointer that points to the current element in the array.
- When you create a new array, the current pointer is initialized to point to the first element in the array.
- Calling **current($array_name)**returns the first element.

```php
var_dump(current($fruits)); // banana
```

Array pointer functions

- Current, next, reset, end, prev

```php
$fruits = ['banana', 'apple',"Kiwi","Orange"];
$found = in_array('banana', $fruits); // true
var_dump(current($fruits)); // banana
var_dump(next($fruits)); //apple
var_dump(next($fruits)); // kiwi
var_dump(current($fruits)); // kiwi
var_dump(reset($fruits)); //banana
var_dump(end($fruits)); //Orange
var_dump(prev($fruits)); //Kiwi
```

Array walk

- Let you want to modify every element in an array, use the
- Array_walk function to do this.

```php
function print_fruits($value){
    echo "$value <br/>";
}
$fruits = ['banana', 'apple',"Kiwi","Orange"];
array_walk($fruits,"print_fruits");
```

banana
apple
Kiwi
Orange

Array merge, chunk

- Convert an associative array into an indexed array.

```php
$a=array(5=>"banana",22=>"Kiwi");
var_dump(array_merge($a));
```

```
array (size=2)
  0 => string 'banana' (length=6)
  1 => string 'Kiwi' (length=4)
```

- Array_chunck: split an array into smaller chunks

```php
$input_array = array('a', 'b', 'c', 'd', 'e');
$output_array = array_chunk($input_array, 2);

var_dump($output_array);
```

```
array (size=3)
  0 =>
    array (size=2)
      0 => string 'a' (length=1)
      1 => string 'b' (length=1)
  1 =>
    array (size=2)
      0 => string 'c' (length=1)
      1 => string 'd' (length=1)
  2 =>
    array (size=1)
      0 => string 'e' (length=1
```

Array navigation

- To loop on two arrays at once (assuming both have the same length)

```php
$instructors = ["Eng. Shery", "Noha", "Andrew"];
$courses = ['Admin', 'PHP', 'Node'];
$result=array_map(function($instructor, $course) {
    return "$instructor teaches $course <br>";
    }, $instructors, $courses);

var_dump($result);
```

```
array (size=3)
  0 => string 'Eng. Shery teaches Admin <br>' (length=29)
  1 => string 'Noha teaches PHP <br>' (length=21)
  2 => string 'Andrew teaches Node <br>' (length=24)
```

Array navigation

- Combine two arrays into one key-value pair array

```php
$combinedArray = array_combine($instructors,$courses);
```

```
array (size=3)
  'Eng. Shery' => string 'Admin' (length=5)
  'Noha' => string 'PHP' (length=3)
  'Andrew' => string 'Node' (length=4)
```

Array navigation

- Remove empty values: use array_filter()

```php
$my_array = [1,90,2,null,3,'',55,[],5,6,7,8,""];
$non_empties = array_filter($my_array);
var_dump($non_empties);
```

```
array (size=9)
  0 => int 1
  1 => int 90
  2 => int 2
  4 => int 3
  6 => int 55
  8 => int 5
  9 => int 6
  10 => int 7
  11 => int 8
```

Array navigation

- array_intersect_key : Computes the intersection of arrays using keys for comparison

```php
$array1 = array('blue'  => 1, 'red'  => 2, 'green'  => 3, 'purple' => 4);
$array2 = array('green' => 5, 'blue' => 6, 'yellow' => 7, 'cyan'   => 8);

var_dump(array_intersect_key($array1, $array2));
```

```
array (size=2)
  'blue' => int 1
  'green' => int 3
```

- Use it to pass certain keys.

```php
$arr = array('a' => 123, 'b' => 213, 'c' => 321);
$allowed = array('b', 'c');
print_r(array_intersect_key($arr, array_flip($allowed)));
```

```
Array ( [b] => 213 [c] => 321 )
```

Count elements in an array

- Count, sizeof, array_Count_values

```php
$students=Array("Ali","Ahmed","Mostafa","Omar","Ahmed");
var_dump(count($students)); // 5
var_dump(sizeof($students)); // 5
var_dump(array_count_values($students)); // count the occurance of each item
```

```
array (size=4)
  'Ali' => int 1
  'Ahmed' => int 2
  'Mostafa' => int 1
  'Omar' => int 1
```

Convert array into scalar

- Convert an associative array to a scalar variables

```php
$info=["username"=>"Noha","email"=>"nshehab@iti.gov.eg","track"=>"Application"];
extract($info);
echo $username." ".$email." ".$track;
```

```
Noha nshehab@iti.gov.eg Application
```

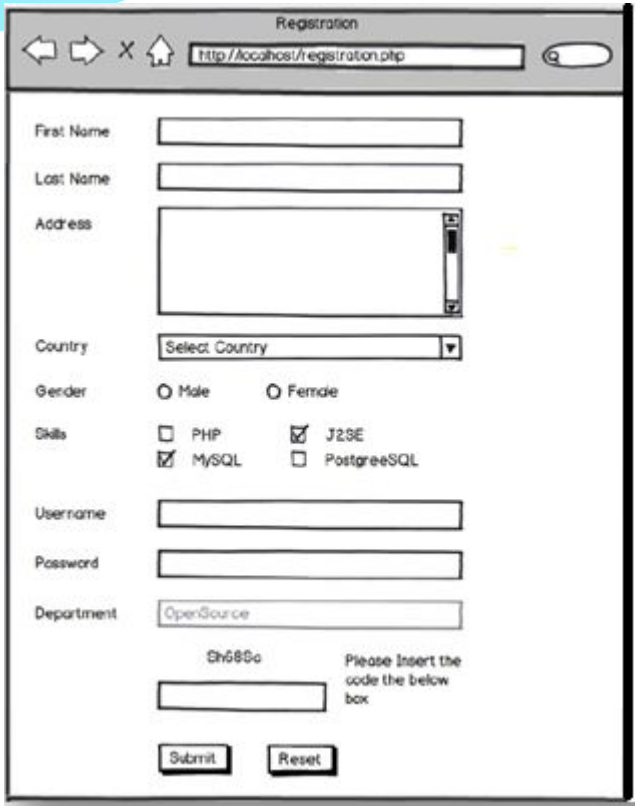
- Convert an indexed array into scalar variable you must provide the scalar variable

```php
$info = array('coffee', 'brown', 'caffeine');
// Listing all the variables
list($drink, $color, $power) = $info;
echo "$drink is $color and $power makes it  special.";
```

```
coffee is brown and caffeine makes it special.
```

Lab 02

- Add the server-side validation to the form fields. (firstname,lastname,email,gender)
- When you submit the data, Save it the customer.txt file.



Lab 02

- Retrieve all the records stored in the file and display them in a table.
- **Bonus: Add the delete button in the table, when you click it, record deleted from the table and file also**

Thanks ^^

Noha Shehab
nshehab@iti.gov.eg