File Handling

File Handling

- PHP offers essential functions for easy file handling in web applications, covering tasks like creation, reading, uploading, and editing.
- File manipulation in PHP is streamlined, with dedicated functions ensuring efficient operations such as opening, processing, and modifying files.
- Simplify file-related tasks in PHP through specialized functions, whether creating files, reading content, or handling uploads in web development.

File Handling

- PHP File Handling
  - File Read
  - Write
  - Close
  - File upload
  - Parsing CSV File
  - Parsing JSON File

File Handling

- File Read: PHP readfile() Function
  - The readfile() function reads a file and writes it to the output buffer.

```
<!DOCTYPE html>
<html>
<body>
<?php
echo readfile("bit.txt");
?>
</body>
</html>
```

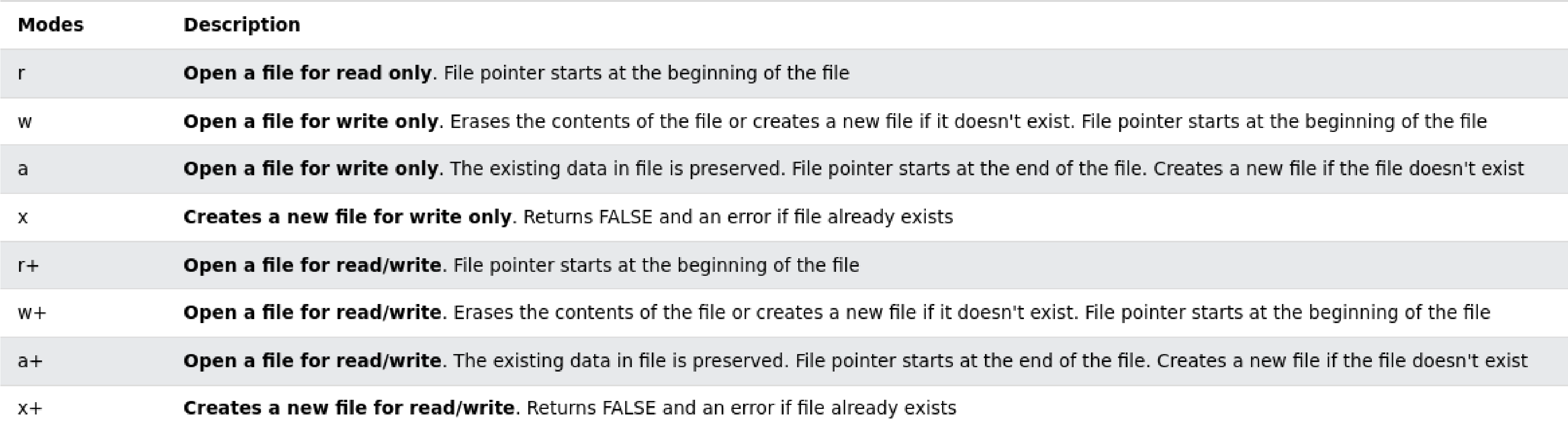
File Handling

- Open/Close
  - Opt for fopen() for versatile file opening, offering more options than readfile().
  - Use fread() with the file name and maximum byte parameters for efficient file reading.
  - The fclose() function is used to close an open file.

```
<!DOCTYPE html>
<html>
<body>
<?php
$myfile = fopen("bit.txt", "r") or die("Unable to open file!");
echo fread($myfile,filesize("bit.txt"));
fclose($myfile);
?>
</body>
</html>
```

File Handling

- Modes for opening file



Modes	Description
r	**Open a file for read only**. File pointer starts at the beginning of the file
w	**Open a file for write only**. Erases the contents of the file or creates a new file if it doesn't exist. File pointer starts at the beginning of the file
a	**Open a file for write only**. The existing data in file is preserved. File pointer starts at the end of the file. Creates a new file if the file doesn't exist
x	**Creates a new file for write only**. Returns FALSE and an error if file already exists
r+	**Open a file for read/write**. File pointer starts at the beginning of the file
w+	**Open a file for read/write**. Erases the contents of the file or creates a new file if it doesn't exist. File pointer starts at the beginning of the file
a+	**Open a file for read/write**. The existing data in file is preserved. File pointer starts at the end of the file. Creates a new file if the file doesn't exist
x+	**Creates a new file for read/write**. Returns FALSE and an error if file already exists

File Handling

- Write/Create
  - In PHP, the fopen() function serves a dual purpose—it not only opens files but also creates them if they don't exist.
  - This occurs when the file is opened for writing ('w') or appending ('a').

```
<?php
$myfile = fopen("newBit.txt", "w") or die("Unable to open file!");
$txt = "Hello! Welcome to the last class of this Semester.\n";
fwrite($myfile, $txt);
$txt = " Hope we will meet in upcoming Semester...!\n";
fwrite($myfile, $txt);
fclose($myfile);
?>
```

File Handling

- File upload
  - Uploading files with PHP is straightforward, but exercise caution due to potential risks.
  - To enable file uploads, check and set the "file_uploads" directive to "On" in the "php.ini" file.

File Handling

- File upload : Following steps in our context
  - Create
    - HTML Form
    - PHP Script for
      - Uploading File
      - Checking if File Already Exists
      - Limiting File Size
      - Limiting File Type

- Our demo files are organized as:
  - file
    - upload_file.html
    - Upload.php

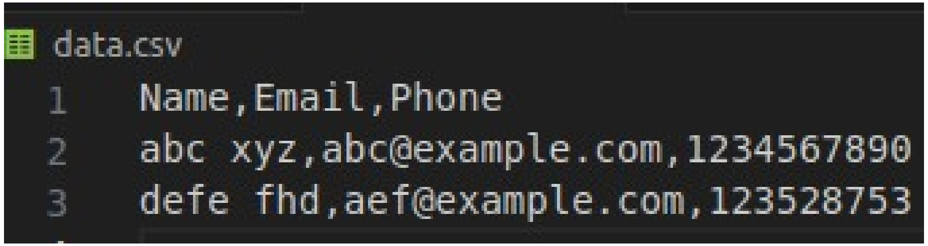
File Handling

- **Parsing CSV File**
  - CSV (Comma-Separated Values) files are a popular format for storing and exchanging tabular data.
  - In a CSV file, each line represents a row of data, and the values in each row are separated by commas or other delimiters (such as semicolons or tabs).
- **Key characteristics of CSV files:**
  - Format: Data is organized in a plain text format, with each field separated by a delimiter (commonly a comma).
  - Structure: Typically, the first row contains headers that describe the content of each column.
  - Common Use: Widely used for importing and exporting data between databases, spreadsheets, and various applications.
  - Compatibility: CSV files can be easily created and read by many software tools, making them a versatile choice for data interchange.

File Handling

- **Parsing CSV File**

```php
<?php
$filename = 'data.csv';
// Check if the file exists and is readable
if (($handle = fopen($filename, 'r')) !== false) {
    echo "<table border='1'>";
    // Read each line from the file
    while (($data = fgetcsv($handle, 1000, ',')) !== false) {
        echo "<tr>";
        // Output each column value
        foreach ($data as $value) {
            echo "<td>$value</td>";
        }
        echo "</tr>";
    }
    echo "</table>";
    // Close the file handle
    fclose($handle);
} else {
    echo "Cannot open the file.";
}
?>
```



data.csv

```
Name,Email,Phone
abc xyz,abc@example.com,1234567890
defe fhd,aef@example.com,123528753
```

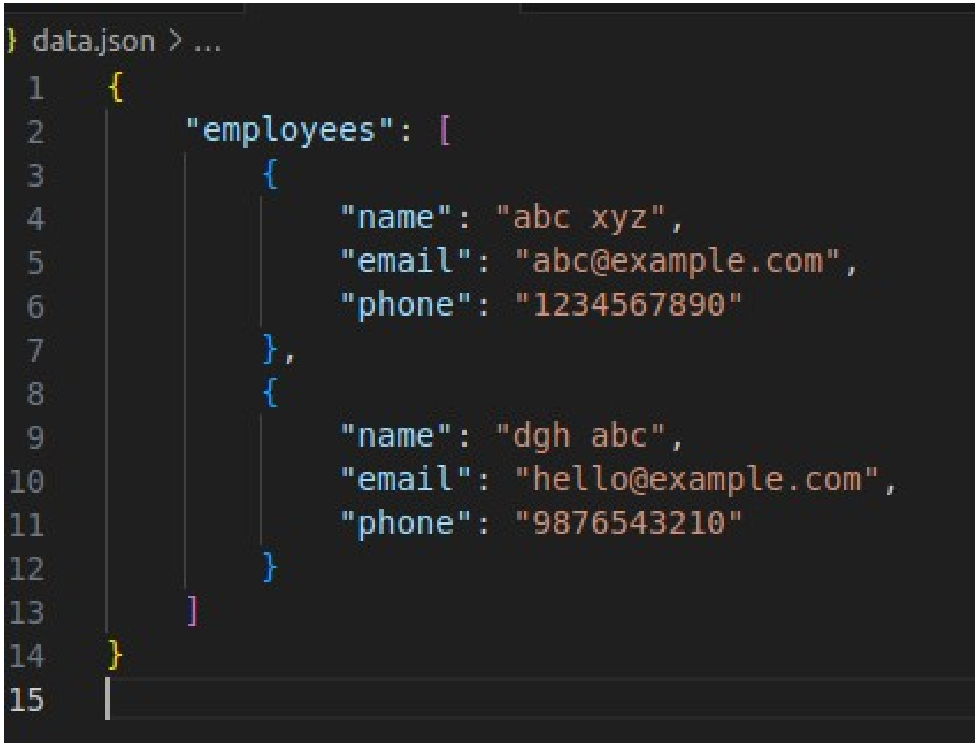
File Handling

- ***Parsing JSON File***
  - *JSON (JavaScript Object Notation) files are a lightweight data-interchange format that is easy for humans to read and write, and easy for machines to parse and generate.*
  - *JSON is often used to transmit data between a server and a web application, as well as to store configuration data.*
- *Here are some key points about JSON files:*
  - *Format: JSON files use a simple text format to represent data as key-value pairs. Data is structured as objects (unordered sets of key-value pairs) or arrays (ordered lists of values).*
  - *Syntax: JSON syntax is similar to JavaScript object notation, making it easy for both humans and machines to understand. It includes data types like strings, numbers, booleans, arrays, objects, and null.*
  - *Common Use Cases:*
    - *Configuration files in web applications.*
    - *Data interchange format in web APIs.*
    - *Storing and exchanging structured data.*
  - *File Extension: JSON files typically have a ".json" file extension.*

File Handling

- **Parsing JSON File**

```
<?php
$filename = 'data.json';
// Check if the file exists and is readable
if (file_exists($filename) && is_readable($filename)) {
    // Read file contents
    $jsonContent = file_get_contents($filename);
    // Decode JSON data to PHP array
    $data = json_decode($jsonContent, true);
    if ($data !== null) {
        echo "<table border='1'>";
        foreach ($data['employees'] as $employee) {
            echo "<tr>";
            echo "<td>{$employee['name']}</td>";
            echo "<td>{$employee['email']}</td>";
            echo "<td>{$employee['phone']}</td>";
            echo "</tr>";
        }
        echo "</table>";
    } else {
        echo "Invalid JSON format.";
    }
} else {
    echo "Cannot open the file.";
}
?>
```



} data.json > ...

```
{
    "employees": [
        {
            "name": "abc xyz",
            "email": "abc@example.com",
            "phone": "1234567890"
        },
        {
            "name": "dgh abc",
            "email": "hello@example.com",
            "phone": "9876543210"
        }
    ]
}

```