CSV (comma-separated values) file

In PHP, you can handle CSV files using various built-in functions and libraries. Here's a basic guide on how to work with CSV files in PHP:

(1) Reading CSV Files:

You can read data from a CSV file using the fgetcsv() function or by reading the file line by line and then splitting each line into an array. Here's an example using fgetcsv():

php

```
$file = fopen('data.csv', 'r');

while (($row = fgetcsv($file)) !== false) {
  // $row is an array containing the CSV data for one row
  print_r($row);
}

fclose($file);
```

(2) Writing to CSV Files:

To write data to a CSV file, you can use the fputcsv() function, which formats an array into a CSV line and writes it to the file. Here's an example:

php

```
$data = array('John Doe', 'john@example.com', 'New York');
$file = fopen('data.csv', 'a'); // Open for appending

fputcsv($file, $data);

fclose($file);
```

Parsing CSV Files using str_getcsv():

If you have CSV data as a string, you can use str_getcsv() to parse it into an array:

php

```
$csvData = "John Doe,john@example.com,New York";
```

```
$dataArray = str_getcsv($csvData);

print_r($dataArray);
```

(3) Using fgetcsv() with Custom Delimiters and Enclosures:

If your CSV file uses custom delimiters or enclosures, you can specify them as arguments to fgetcsv():

php

```
$file = fopen('data.csv', 'r');


while (($row = fgetcsv($file, 0, ';', '"')) !== false) {

   // Use ';' as delimiter and '"' as enclosure

   print_r($row);

}


fclose($file);
```