PHP - File Handling

In PHP, a file is a resource object, from which data can be read or written to in a linear fashion. The term "file handling" refers to a set of functions in PHP that enable read/write operations on disk files with PHP code.

A file object is classified as a **stream**. Any resource on which linear read/write operations are done is a stream. Other stream-like objects are the TCP sockets, standard input stream, i.e., a system keyboard represented by "php://stdin", the standard output stream represented by "php://stdout", and the error stream "php://stderr".

Note – Tthe constants STDIN, STDOUT, and STDERR stand for the respective standard streams.

Although PHP is regarded as a server-side scripting language for developing web applications, PHP also has a command-line interface to perform console IO operations.

Example

The readline() function in PHP accepts the user input from a standard keyboard, and echo/print statements render the output on the console.

```
<?php
   $str = readline("Type something:");
   echo $str;
?>
```

It will produce the following **output** –

```
C:\xampp\php>php hello.php
Type something: Are you enjoying this PHP tutorial?
Are you enjoying this PHP tutorial?
```

Example

We can obtain the same effect by reading the input from "php://stdin" and outputting it to "php://stdout".

```
<?php
   $f = fopen("php://stdin", "r");
```

```
   echo "Type something: ";

   $str = fgets($f);
   $f1 = fopen("php://stdout", "w");

   fputs($f1, $str);
?>
```

Here, the fopen() function is used to open the **stdin** stream for reading and the **stdout** stream for writing.

Example

PHP supports a variety of stream protocols for stream related functions such as fopen(), file_exists(), etc. Use php_get_wrappers() function to get a list of all the registered wrappers.

</> Open Compiler

```
<?php
   print_r(stream_get_wrappers());
?>
```

It will produce the following **output** –

```
Array
(
   [0] => php
   [1] => file
   [2] => glob
   [3] => data
   [4] => http
   [5] => ftp
   [6] => zip
   [7] => compress.zlib
   [8] => compress.bzip2
   [9] => https
   [10] => ftps
```

```
    [11] => phar
)
```

The streams are referenced as "scheme://target". For instance, the file stream is "file://xyz.txt".

The input data from the console is stored in the computer's main memory (RAM) until the application is running. Thereafter, the memory contents from RAM are erased.

We would like to store it in such a way that it can be retrieved whenever required in a persistent medium such as a disk file. Hence, instead of the standard streams (keyboard for input and the display device for output), we will use the disk files for reading the data, and destination for storing the data.

In addition to the read and write modes as used in the above example (IO operations with standard streams), the file stream can be opened in various other modes like "r+" and "w+" for simultaneous read/write, "b" for binary mode, etc.

To open a disk file for reading and obtain its reference pointer, use the fopen() function.

```
$handle = fopen('file://' . __DIR__ . '/data.txt', 'r');
```

The "file://" scheme is the default. Hence, it can be easily dropped, especially when dealing with local files.

Note – It is always recommended to close the stream that was opened. Use the fclose() function for this purpose.

```
fclose($handle);
```

PHP has several built-in functions for performing read/write operations on the file stream. In the subsequent chapters, we shall explore the filesystem functions.