File Handling in PHP

Opening and Closing Files

Files are opened in PHP using the **fopen** command. The command has two parameters.

- File to be opened
- mode in which to open the file.

Syntax

$filehandle=fopen(filename, mode [use_include_path [, zcontext]])

Where ***filename*** is the name of the file to be opened, ***mode*** indicates how to open the file, ***use_include_path*** may be set to 1 or TRUE to specify path to be searched for the file, ***zcontext*** holds an optional file context.

File Modes

The following table shows the different modes the file may be opened in.

Mode	Description
r	Read Only mode, with the file pointer at the start of the file.
r+	Read/Write mode, with the file pointer at the start of the file.
W	Write Only mode. Truncates the file (effectively overwriting it). If the file doesn't exist, fopen will attempt to create the file.
w+	Read/Write mode. Truncates the file (effectively overwriting it). If the file doesn't exist, fopen will attempt to create the file.
a	Append mode, with the file pointer at the end of the file. If the file doesn't exist, fopen will attempt to create the file.
a+	+ Read/Append, with the file pointer at the end of the file. If the file doesn't exist, fopen will attempt to create the file
x	Create and open for writing only. If the file already exists, fopen() will fail by returning FALSE and generating an error. If the file does not exist, attempt to create it.
x+	Create and open for reading and writing. If the file already exists, fopen() will fail by returning FALSE and generating an error. If the file does not exist, attempt to create it.

Note: The *mode* may also contain the letter 'b'. This is useful only on systems which differentiate between binary and text files (i.e.Windows. It's useless on Unix/Linux). If not needed, it will be ignored.

First create a data.txt file by opening a notepad and type the following

This
Is
My
Data file

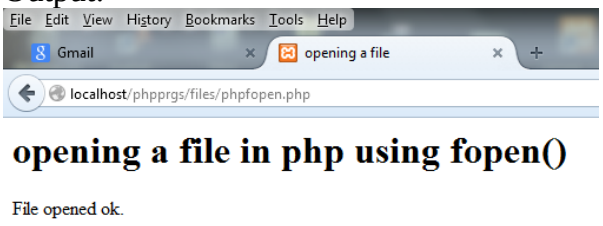
You can read the content of this file by the following php coding

```
<?php
$handle=fopen("data.txt","r");
.
.
.
?>
```

If the open operation fails , fopen returns FALSE.

```
<html>
<head>
<title>opening a file</title>
</head>
<body>
<h1> opening a file in php using fopen()</h1>
<?php
$handle=fopen("data.txt","r");
if($handle)
{
echo "File opened ok.";
}
?>
</body>
</html>
```

Output:



Using feof for looping over all the lines in file

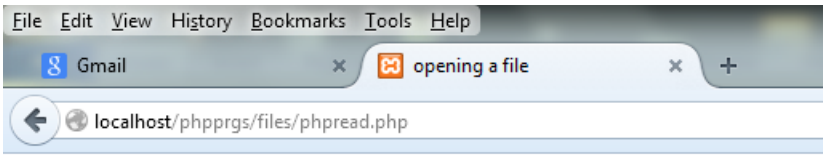
When there are multiple lines in a file, we would like to read all the lines, i.e till the end of the file. There is function called feof() which do this . it checks whether reading has reached end of the file and returns true if it reached.
fgets() function is used to get a string of text from a file; syntax is

fgets (handle [, length])

example

```
<?php
$handle=fopen("data.txt","r");
while (!feof($handle))
{
$x=fgets($handle);
.
.
}
.
.
?>
```

```
<html>
<head>
<title>opening a file</title>
</head>
<body>
<h1> opening a file in php using fopen()</h1>
<?php
$handle=fopen("data.txt","r");
while (!feof($handle))
{
$x=fgets($handle);
echo $x,"<br>";
}
?>
</body>
</html>
```



opening a file in php using fopen()

This
Is
My
Data file

Closing a file

The file can be closed with the command

fclose($filehandle);

This frees up the resources connected with that file.

Reading from a file character by character with fgetc()

Syntax
fgetc($filehandle);

This function returns the character read.
To read an individual character from file.txt :

```
<?php
$handle=fopen("file.txt","r");
$char=fgetc($handle)
.
..
}
?>
```

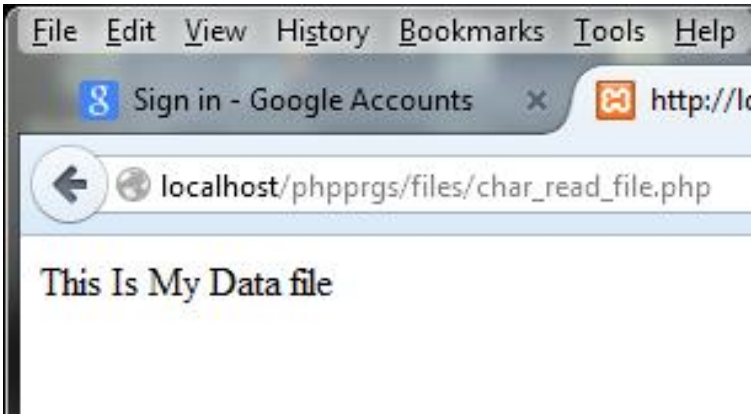
To loop over all the characters in the file, you can put the preceding statement in the condition of a while loop- when fgetc returns FALSE, there are no more characters to read:

```
<?php
$handle=fopen("file.txt","r");
while($char=fgetc($handle))
{
.
..
}
?>
```

And you can echo each character as you read:

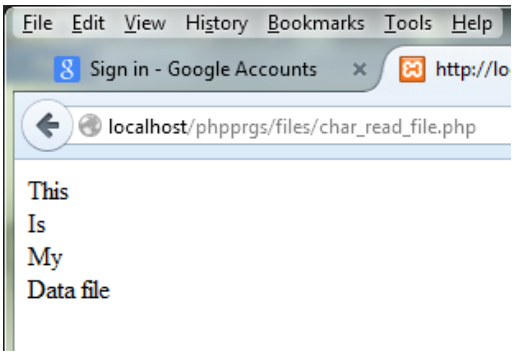
```
<?php
$handle=fopen("data.txt","r");
while($char=fgetc($handle))
{
echo "$char";
}
?>
```

When a newline character is read from the file it is simply sent to the browser, which doesn't display newline characters and do nothing. You have to convert them to
 element instead. In the above program the '\n' is not replaced. So you can see the effect in the output.



In replacing the '\n' with
 as below you can see the difference.

```
<?php
$handle=fopen("data.txt","r");
while($char=fgetc($handle))
{
if($char=="\n")
{
$char="<br>";
}
echo "$char";
}
fclose($handle);
?>
```



Reading the whole file at once with file_get_contents

You can read the entire contents of the file with the file_get_contents function:
Syntax
file_get_contents(filename [, use_include_path [, context [, offset [, maxlen]]]])

filename is the name of the file on which operation is carried over
use_include_path is set to TRUE if you want to search PHP's include path
context is a context for the operation
offset is the offset into the file at which to start reading
maxlen is the maximum length of data to read.

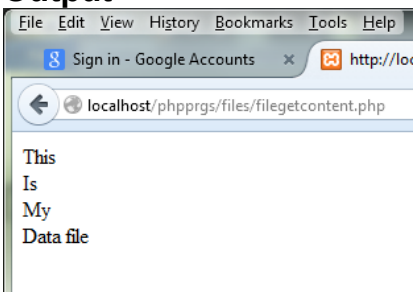
In the below example the entire content of a file is read into a variable $text at once:

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

The code to convert all newlines to
 is str_replace.

```
<?php
$text=file_get_contents("data.txt");
$text1=str_replace("\n","<br>",$text);
echo $text1;
?>
```

Output



Reading a file into an array with file

You can use the file function to read a file into an array all at once. Each line becomes an element in the array.
Syntax
file (filename [, use_include_path [, context]])

filename is the name of the file on which operation is carried over
use_include_path is set to TRUE if you want to search PHP's include path
context is a context for the operation

In the below example the entire content of a file is read into an array $array1 at once:

```
<?php
$array1=file("data.txt");
.
.
}
?>
```

After which you can read the data in array as usual using foreach statement as below:

```
<?php
$array1=file("data.txt");
foreach($array1 as $line)
{
echo $line, "<br>";
}
?>
```

Checking a file existence:

You can check whether a file exists or not by using file_exists command
Syntax
file_exists(filename)

```
<?php
if(file_exists("data.txt"))
{
$array1=file("data.txt");

foreach($array1 as $line)
{
echo $line, "<br>";
}
else
echo "file doesnot exists";
?>
```

Getting the size of the file

The file size can be received using filesize function
Syntax
filesize(filename)

Deleting a file

You can delete a file using unlink function
Syntax
unlink(filename [,context])

writing to a file with fwrite

syntax
fwrite(handle, string [, length])

```
<?php
$handle=fopen("data1.txt","w");
$text="here\n is \n the \ntext";
fwrite($handle,$text);

?>
```

Appending to a file with fwrite

By changing the file mode to “a” we can append the existing data file as follows

```
<?php
$handle=fopen("data1.txt","a");
//$text="here\n is \n the \ntext";
$text1="this is appending";
fwrite($handle,$text1);

?>
```