PHP forms

This lecture covers

- Getting data from forms
- html form + php action on the same page
  - `$_SERVER['PHP_SELF']`
  - maintain form status
- page redirection - the `header()` function
- Upload files
- Files
  - creation and deletion
  - opening and closing
  - reading …
- `print_r` function

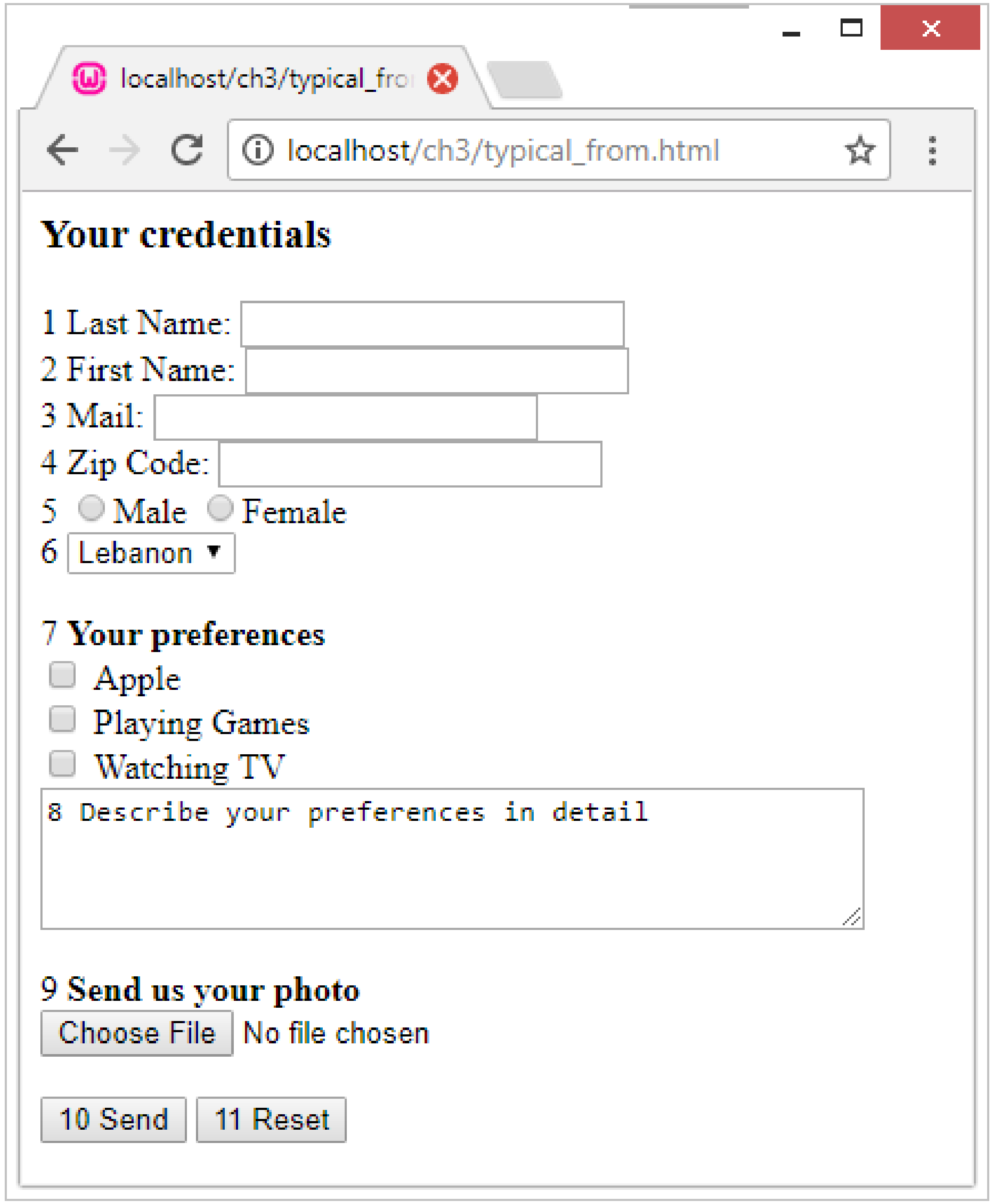
This lecture covers

- **Getting data from forms**
- html form + php action on the same page
    - `$_SERVER['PHP_SELF']`
    - maintain form status
- page redirection - the `header()` function
- Upload files
- Files
    - creation and deletion
    - opening and closing
    - reading …
- `print_r` function

Getting data from forms

- what happens when the user clicks the submit button?
  - an HTTP request is sent to the server to the script named by the action attribute of the `<form>`
  - request contains all associations between field names and their values
    - in the body of the HTTP request if the `POST` method is used
    - in the URL if it is the `GET` method

Typical form


localhost/ch3/typical_from.html

Your credentials

1 Last Name:
2 First Name:
3 Mail:
4 Zip Code:
5 Male Female
6 Lebanon

7 **Your preferences**
Apple
Playing Games
Watching TV
8 Describe your preferences in detail

9 **Send us your photo**
Choose File No file chosen

10 Send 11 Reset

Unique values

- are from the form fields in which the user
  - can only enter a value, such as a text, or
  - can only make one choice (radio button, select list)
- since PHP 4.1
  - values → superglobal associative arrays : `$_POST` or `$_GET`
  - array keys = names associated with fields by the name attribute

Getting values

- to simplify → get value in scalar variable, then reuse
  - `$name= $_POST["name"];`
  - `$level=$_POST["level"];`
- W3C (World Wide Web Consortium)
  - consider removing the `get` value for the method attribute of forms in HTML or xHTML

Multiple Values

- some fields → allow the entry of several values under the same name
  - group of checkboxes with the same name attribute (check one or more)
  - selection list with multiple attribute = "multiple"
  - give the same name to different text entry elements, less interest
- table that is retrieved on the server side ➔ must follow the name of the component of brackets, as to create a variable of type array
- example
  - `Blue :<input type="checkbox" name="choice[]" value="blue" />`
  - `White :<input type="checkbox" name="choice[]" value="white" />`
- Getting the values
  - `$_POST["choice"][0]` contains the value "blue"
  - `$_POST["choice"][1]` contains the value "white"

This lecture covers

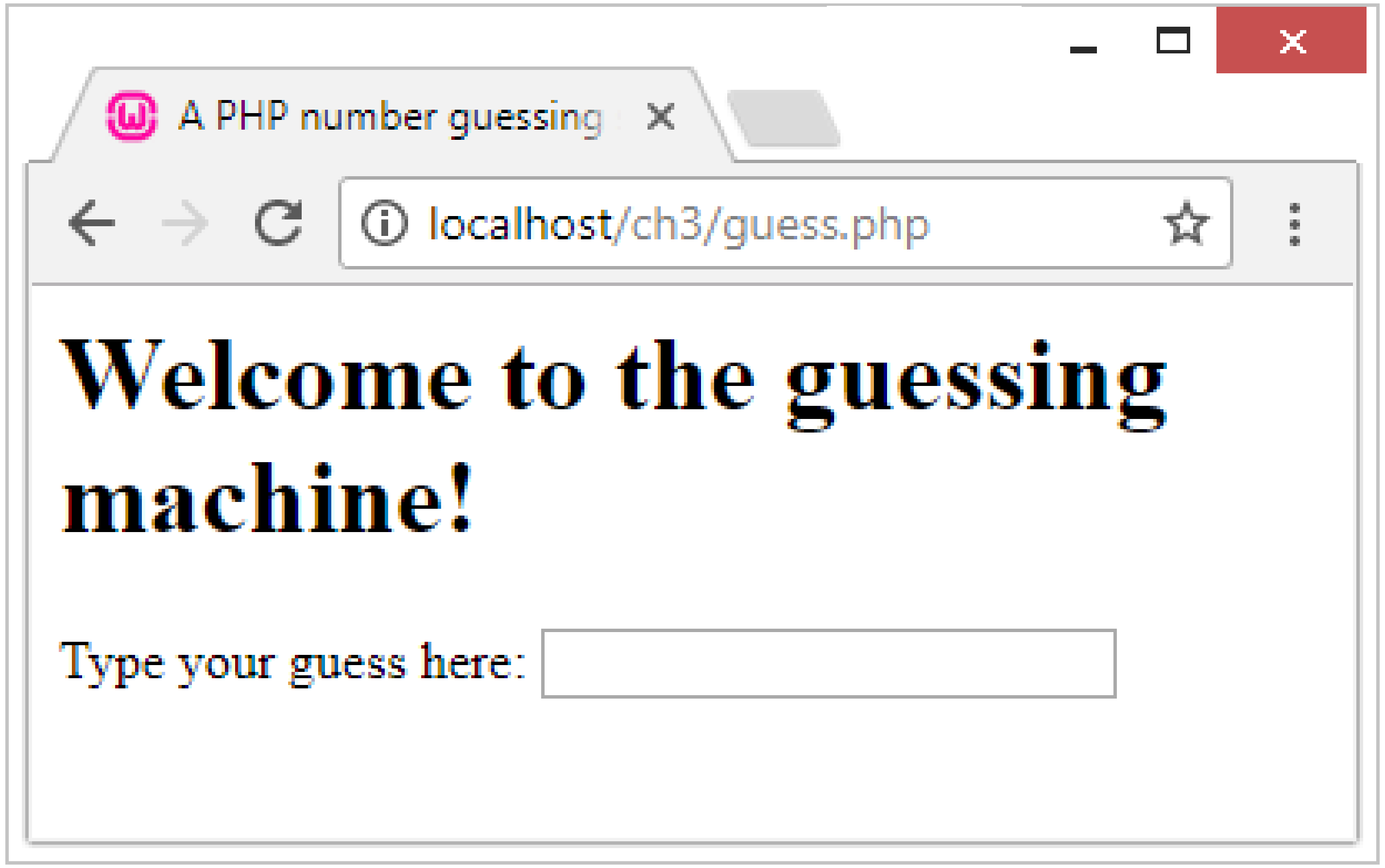
- Getting data from forms
- **html form + php action on the same page**
  - **`$_SERVER['PHP_SELF']`**
  - **maintain form status**
- page redirection - the `header()` function
- Upload files
- Files
  - creation and deletion
  - opening and closing
  - reading …
- `print_r` function

The combination of the HTML form and the PHP action on the same page

```
<form  action="<?php print $_SERVER['PHP_SELF']?>"
      method="post">
```

- `$_SERVER['PHP_SELF']`
  - the action of this script is `$_SERVER ['PHP_SELF']`
  - this variable is equivalent to the name of the current script
  - the action indicates that the script is reloading itself

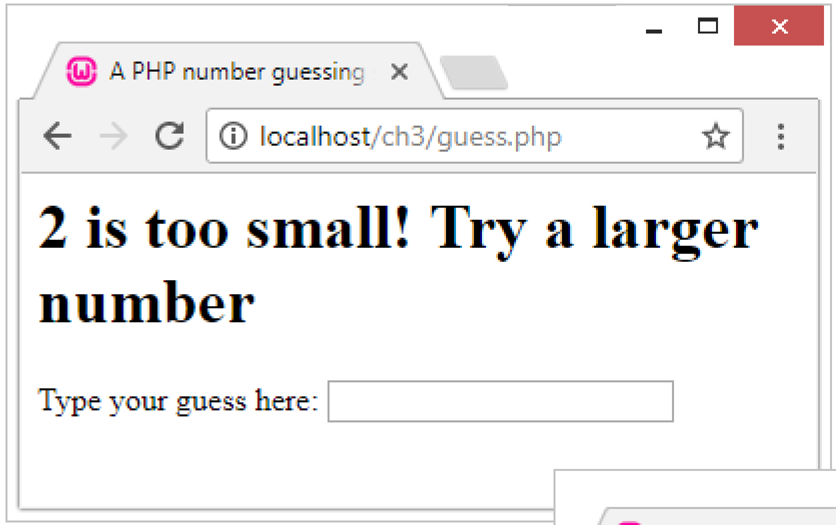
Example: Guessing game



Welcome to the guessing machine!

Type your guess here:

Example: Guessing game

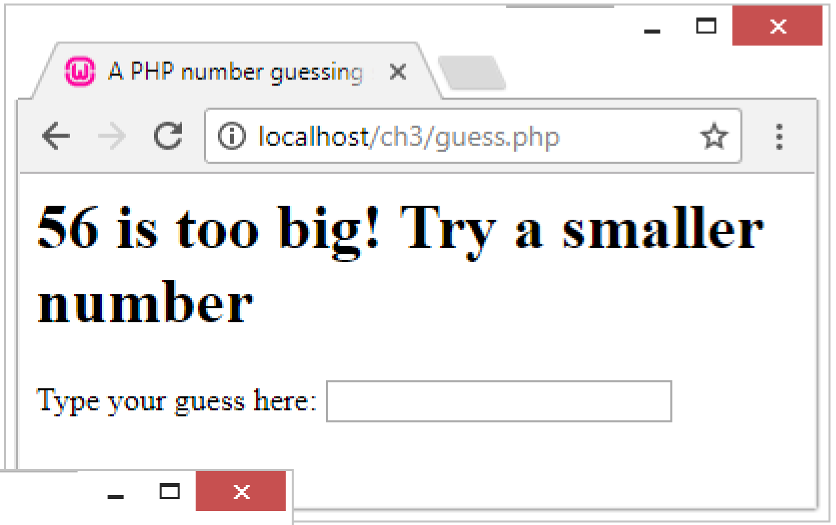


A PHP number guessing

localhost/ch3/guess.php

2 is too small! Try a larger number

Type your guess here:

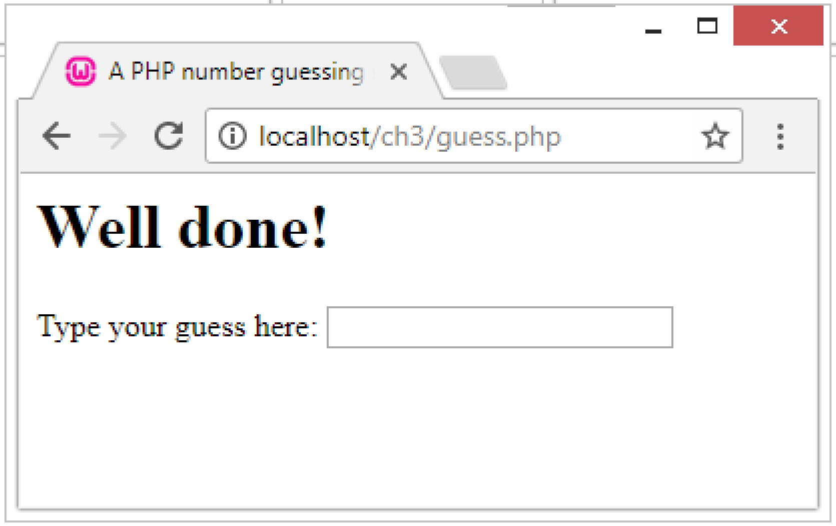


A PHP number guessing

localhost/ch3/guess.php

56 is too big! Try a smaller number

Type your guess here:



A PHP number guessing

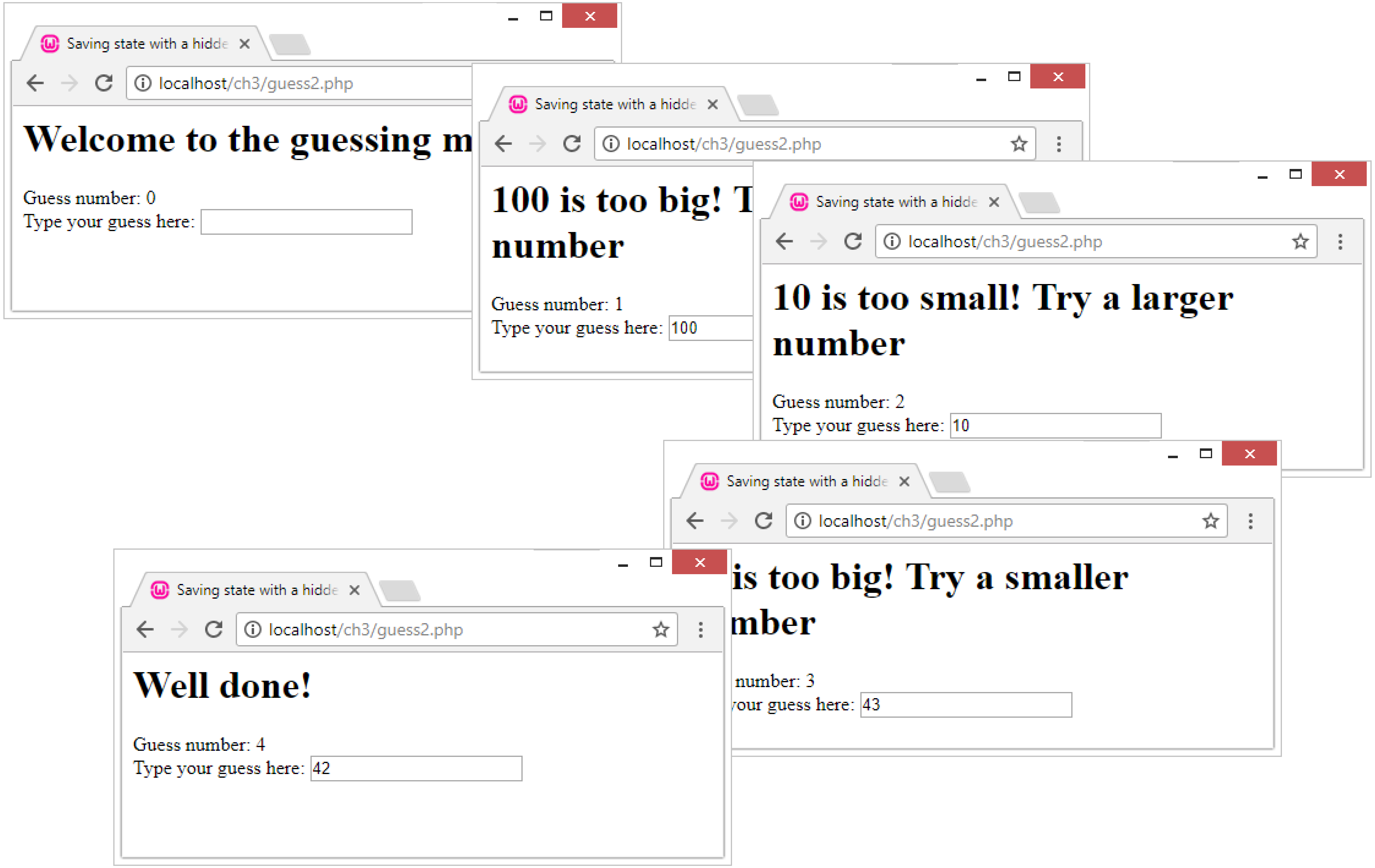
localhost/ch3/guess.php

Well done!

Type your guess here:

Example: Guessing game

```
<?php
            $num_to_guess = 42;
            $message = "";
            if (!isset($_POST[guess])) {
                        $message = "Welcome to the guessing machine!";
            } elseif ($_POST[guess] > $num_to_guess) {
                        $message = "$_POST[guess] is too big! Try a smaller number";
            } elseif ($_POST[guess] < $num_to_guess) {
                        $message = "$_POST[guess] is too small! Try a larger number";
            } else { // must be equivalent
                        $message = "Well done!";
            }
?>
<html>
<head>
            <title>A PHP number guessing script</title>
</head>
<body>
            <h1><?php print $message ?></h1>
            <form action="<?php print $_SERVER[PHP_SELF] ?>" method="POST">
            Type your guess here: <input type="text" name="guess">
            </form>
</body>
</html>
```



Welcome to the guessing m

Guess number: 0
Type your guess here:

100 is too big! T
number

Guess number: 1
Type your guess here: 100

10 is too small! Try a larger number

Guess number: 2
Type your guess here: 10

is too big! Try a smaller
mber

number: 3
our guess here: 43

Well done!

Guess number: 4
Type your guess here: 42

Using Hidden Fields to Save State

- we want to count the number of attempts
  - we use a variable `$num_tries` which is initialized for a first visit to zero
  - this variable is passed in a field hidden in the form, to increment it after each test
- we want to keep the guessed value in the text field after reloading the form
  - we give a default value to the text field "guess"
  - it is the value of this same field already entered
  - `$_POST['guess']`

Example: Guessing game

```
<?php
            $num_to_guess = 42;
            $num_tries = (isset($_POST[num_tries])) ? $_POST[num_tries] + 1 : 0;
            $message = "";
            if (!isset($_POST[guess])) {
                        $message = "Welcome to the guessing machine!";
            } else
                        {if ($_POST[guess] > $num_to_guess) {
                                    $message = "$_POST[guess] is too big! Try a smaller number";
                        } elseif ($_POST[guess] < $num_to_guess) {
                                    $message = "$_POST[guess] is too small! Try a larger number";
                        } else { // must be equivalent
                                    $message = "Well done!";
                        }
                        $guess = $_POST[guess];
            }
?>
<html>
<head><title>Saving state with a hidden field</title>
</head>
<body>
<h1><?php print $message ?></h1>
            Guess number: <?php print $num_tries?>
            <form action="<?php print $_SERVER[PHP_SELF] ?>" method="POST">
            Type your guess here:
            <input type="text" name="guess" value="<?php print $guess?>">
            <input type="hidden" name="num_tries" value="<?php print $num_tries?>">
            </form>
</body>
</html>
```

This lecture covers

- Getting data from forms
- html form + php action on the same page
  - `$_SERVER['PHP_SELF']`
  - maintain form status
- **page redirection - the `header()` function**
- Upload files
- Files
  - creation and deletion
  - opening and closing
  - reading …
- `print_r` function

Redirecting the User

- previous script → major disadvantage
- form is rewritten even if the user correctly guesses
- HTML is hard-coded → difficult to avoid writing the entire page
- solution :
  - redirect the user to another page → congratulations

`header()` function

- when a server script communicates with a client,
  - it must first send headers containing information about the upcoming document
  - you can choose to send other header lines with the PHP function: `header()`
- to call the `header()` function,
  - **ensure that no line has been sent to the browser**
- any display in document,
  - even a line break or outside space to your script tags
  - causes the headers to be sent to the client
- If you intend to use the `header()` function in a script
  - ensure that nothing precedes the PHP code that contains the function call
  - check used libraries

Using `header()` to Send Raw Headers

- assuming → a page "congrats.html"
- edit the script to redirect the user if he guesses correctly

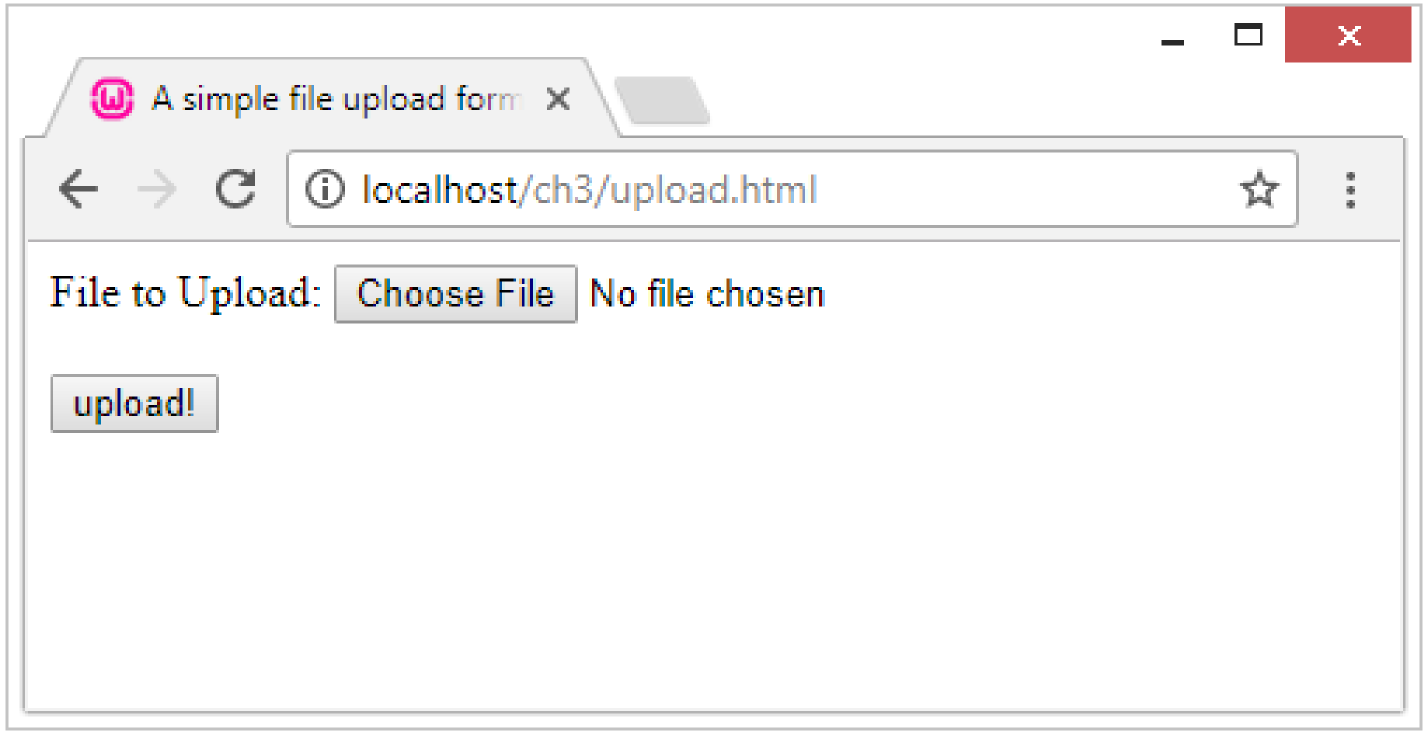
Example: Guessing game

```
<?php
	$num_to_guess = 42;
	$num_tries = (isset($_POST[num_tries])) ? $_POST[num_tries] + 1 : 0;
	$message = "";
	if (!isset($_POST[guess])) {
		$message = "Welcome to the guessing machine!";
	} else
		{if ($_POST[guess] > $num_to_guess) {
			$message = "$_POST[guess] is too big! Try a smaller number";
		} elseif ($_POST[guess] < $num_to_guess) {
			$message = "$_POST[guess] is too small! Try a larger number";
		} else { // must be equivalent
			header("Location: congrats.html");
			exit;
		}
		$guess = $_POST[guess];
	}
?>
```

This lecture covers

- Getting data from forms
- html form + php action on the same page
  - `$_SERVER['PHP_SELF']`
  - maintain form status
- page redirection - the `header()` function
- **Upload files**
- Files
  - creation and deletion
  - opening and closing
  - reading …
- `print_r` function

Upload files



File to Upload: Choose File No file chosen

upload!

Upload files

```
<html>

        <head><title>A simple file upload form</title></head>

<body>

<form action="upload.php"
       enctype="multipart/form-data" method="POST">

        <input type="hidden" name="MAX_FILE_SIZE" value="51200">

        File to Upload: <input type="file" name="fileupload"><br><br>

        <input type="submit" value="upload!">

</form>

</body>

</html>
```

Upload form

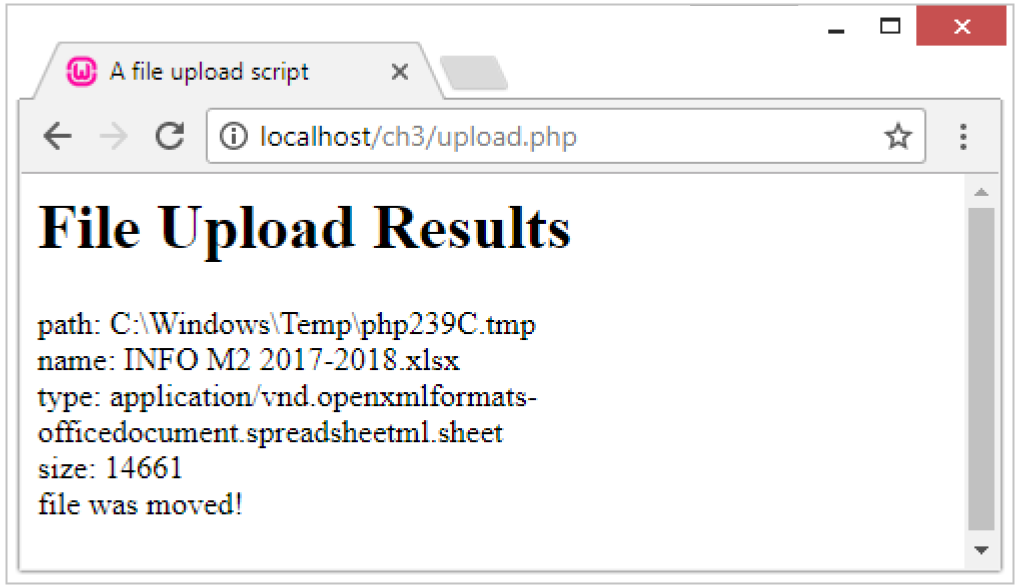
- form
  - attribute method:
    - `post`
  - attribute ENCTYPE :
    - `ENCTYPE="multipart/form-data"`
- hidden field inserted before `file` field
  - must be named `MAX_FILE_SIZE`
  - maximum size in bytes accepted
- this size can not override the maximum `upload_max_filesize` size in the field in your default php.ini file of 2MB
- field to upload the file
  - element `INPUT` of `type="file"`
  - `name="yourchoice"`

PHP code for file

- Information about the uploaded file becomes available to you in the `$_FILES` superglobal, which is indexed by the name of the upload field (or fields) in the form. The corresponding value for each of these keys is an associative array. These fields are described below, using `fileupload` as the name of the form field used for the upload.

Element	Contains	Example
`$_FILES['fileupload']['name']`	Original name of uploaded file	test.gif
`$_FILES['fileupload']['tmp_name']`	Path to temporary file	/tmp/phprDfZvN
`$_FILES['fileupload']['size']`	Size (in bytes) of uploaded file	6835
`$_FILES['fileupload']['type']`	MIME type of uploaded file (where given by client)	image/gif

A Simple File Upload Script

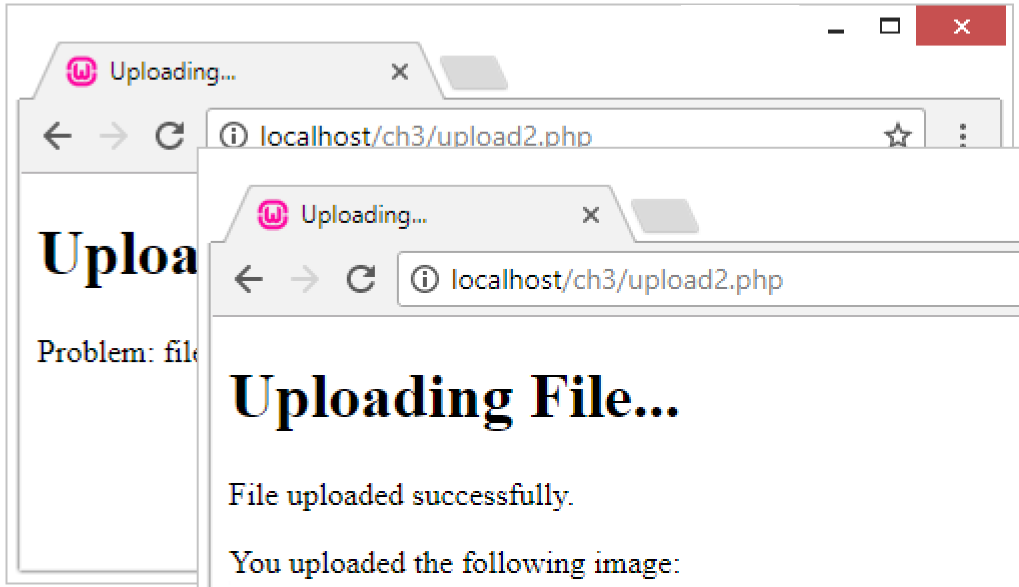


File Upload Results

```
path: C:\Windows\Temp\php239C.tmp
name: INFO M2 2017-2018.xlsx
type: application/vnd.openxmlformats-
officedocument.spreadsheetml.sheet
size: 14661
file was moved!
```

A Simple File Upload Script

```
<html>
<head><title>A file upload script</title></head>
<body>
            <h1>File Upload Results</h1>
            <?php
                        $file_dir = "D:\wamp\www\ch3";

                        foreach($_FILES as $file_name => $file_array) {
                                    print "path: ".$file_array['tmp_name']."<br>\n";
                                    print "name: ".$file_array['name']."<br>\n";
                                    print "type: ".$file_array['type']."<br>\n";
                                    print "size: ".$file_array['size']."<br>\n";

                                    if (is_uploaded_file($file_array['tmp_name'])) {
                                                move_uploaded_file($file_array['tmp_name'],
                                                            "$file_dir/$file_array[name]") or die ("Couldn't copy");
                                                print "file was moved!<br><br>";
                                    }
                        }
            ?>
</body>
</html>
```

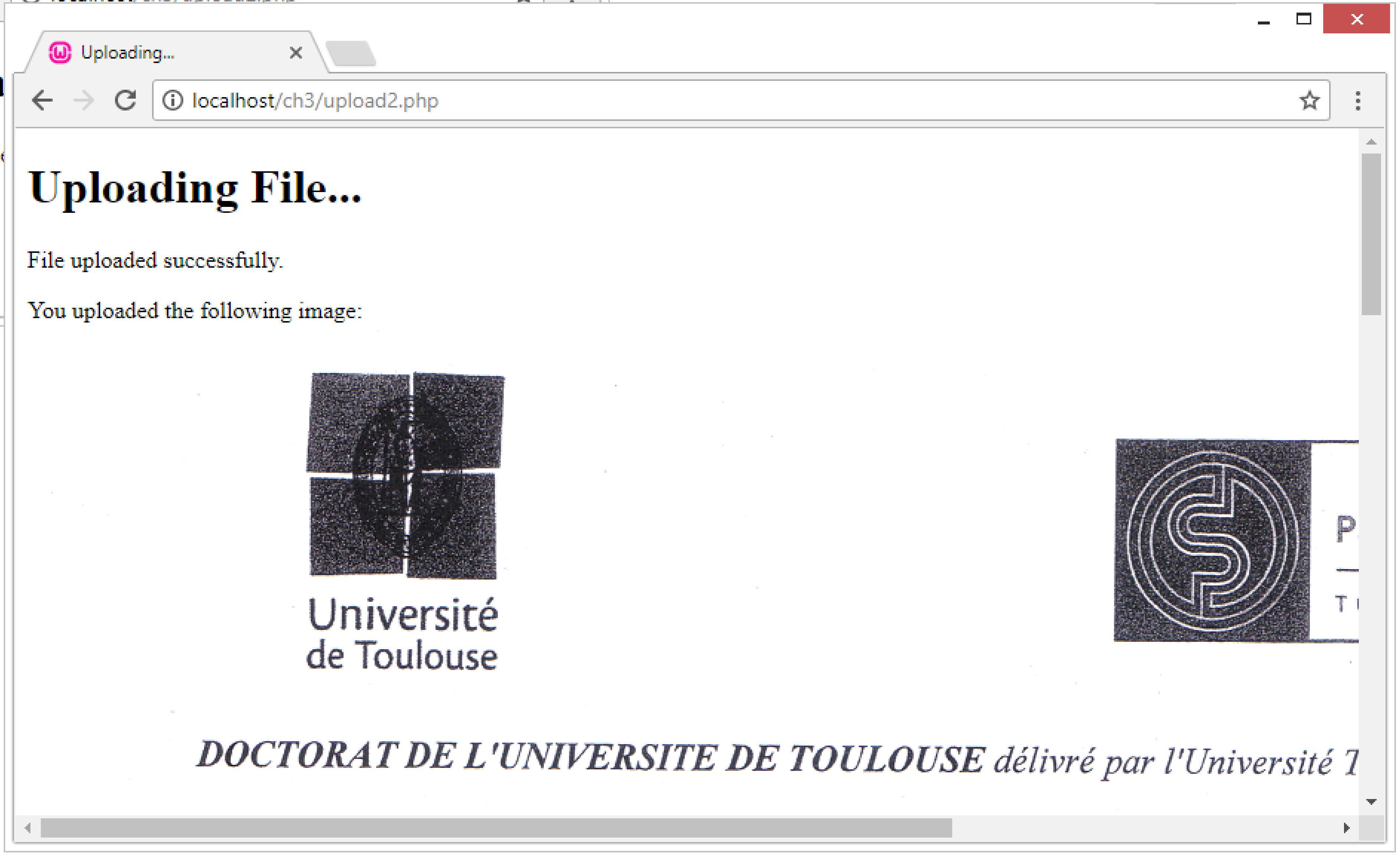


Uploading...

localhost/ch3/upload2.php

Uploa

Problem: fil



Uploading...

localhost/ch3/upload2.php

Uploading File...

File uploaded successfully.

You uploaded the following image:

Université
de Toulouse
DOCTORAT DE L'UNIVERSITE DE TOULOUSE délivré par l'Université

A File Upload Script

```
<!DOCTYPE html>
<html>
<head><title>Uploading...</title></head>
<body>
<h1>Uploading File...</h1>
<?php
        if ($_FILES['fileupload']['error'] > 0)
        {
                echo 'Problem: ';
                switch ($_FILES['fileupload']['error'])
                {
                        case 1: echo 'File exceeded upload_max_filesize.'; break;
                        case 2: echo 'File exceeded max_file_size.'; break;
                        case 3: echo 'File only partially uploaded.'; break;
                        case 4: echo 'No file uploaded.'; break;
                        case 6: echo 'Cannot upload file: No temp directory specified.'; break;
                        case 7: echo 'Upload failed: Cannot write to disk.'; break;
                        case 8: echo 'A PHP extension blocked the file upload.'; break;
                }
                exit;
        }
```

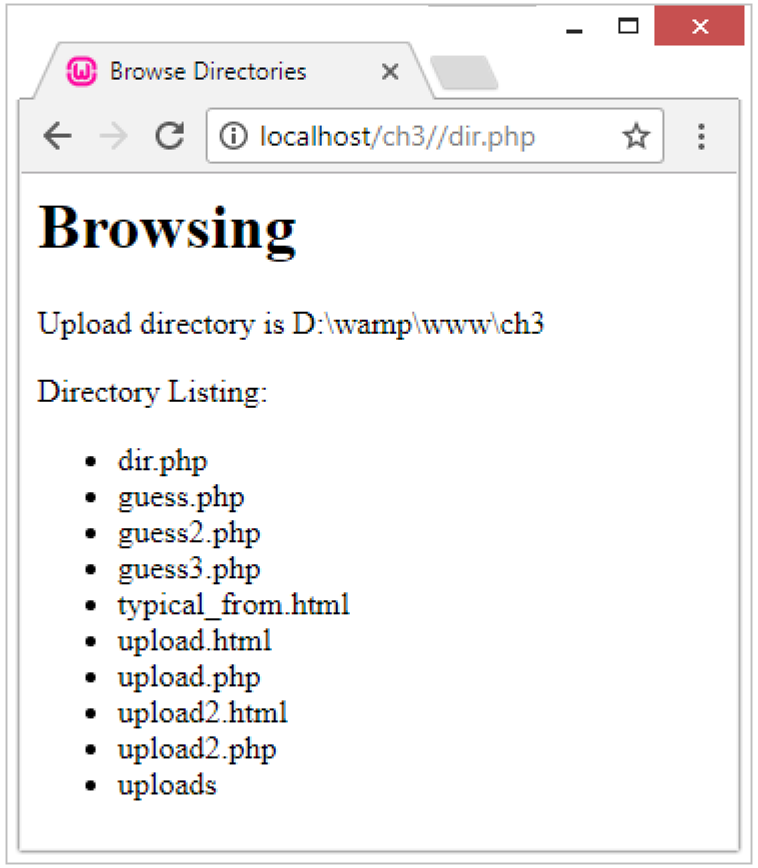
A File Upload Script

```
// Does the file have the right MIME type?
if ($_FILES['fileupload']['type'] != 'image/png')
{
        echo 'Problem: file is not a PNG image.';
        exit;
}

// put the file where we'd like it
$uploaded_file = "D:\\wamp\\www\\ch3\\uploads\\".$_FILES['fileupload']['name'];
if (is_uploaded_file($_FILES['fileupload']['tmp_name']))
{
        if (!move_uploaded_file($_FILES['fileupload']['tmp_name'], $uploaded_file))
        {
                echo 'Problem: Could not move file to destination directory.';
                exit;
        }
}
else
{
        echo 'Problem: Possible file upload attack. Filename: ';
        echo $_FILES['fileupload']['name'];
        exit;
}
echo 'File uploaded successfully.';
```

A File Upload Script

```
       // show what was uploaded
       echo '<p>You uploaded the following image:<br/>';
       echo '<img src="uploads/'.$_FILES['fileupload']['name'].'"/>';
?>
</body>
</html>
```

Browse Directories


Browse Directories

localhost/ch3//dir.php

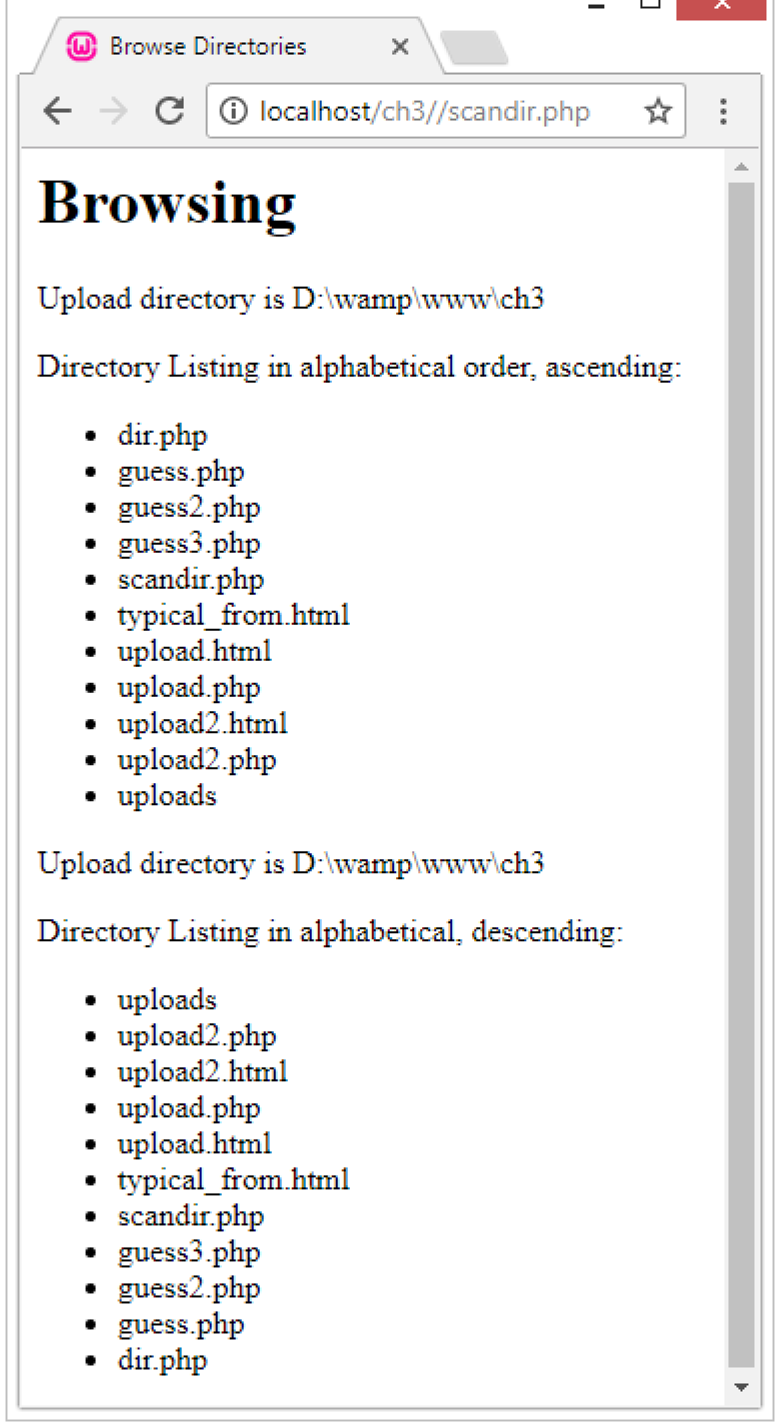
```
Browsing

Upload directory is D:\wamp\www\ch3

Directory Listing:

  • dir.php
  • guess.php
  • guess2.php
  • guess3.php
  • typical_from.html
  • upload.html
  • upload.php
  • upload2.html
  • upload2.php
  • uploads
```

Browse Directories

```
<html>
<head><title>Browse Directories</title></head>
<body>
<h1>Browsing</h1>
<?php
        $current_dir = "D:\\wamp\\www\\ch3";
        $dir = opendir($current_dir);
        echo '<p>Upload directory is '.$current_dir.'</p>';
        echo '<p>Directory Listing:</p><ul>';
        while(false !== ($file = readdir($dir)))
        {
                //strip out the two entries of . and ..
                if($file != "." && $file != "..")
                {
                        echo '<li>'.$file.'</li>';
                }
        }
        echo '</ul>';
        closedir($dir);
?>
```

`scandir()` function

- to sort the filenames in alphabetical order
- The file names in the example above are not sorted in a particular order,
  - so if you need a sorted list, you can use a function called `scandir()`.
- This function can be used to store file names in an array and sort them alphabetically, either ascending or descending, as in the following list.



Browsing

Upload directory is D:\wamp\www\ch3

Directory Listing in alphabetical order, ascending:

- dir.php
- guess.php
- guess2.php
- guess3.php
- scandir.php
- typical_from.html
- upload.html
- upload.php
- upload2.html
- upload2.php
- uploads

Upload directory is D:\wamp\www\ch3

Directory Listing in alphabetical, descending:

- uploads
- upload2.php
- upload2.html
- upload.php
- upload.html
- typical_from.html
- scandir.php
- guess3.php
- guess2.php
- guess.php
- dir.php

scandir() function

```
<html>
<head><title>Browse Directories</title></head>
<body>
<h1>Browsing</h1>
<?php
        $current_dir = "D:\\wamp\\www\\ch3";
        $dir = opendir($current_dir);
        $files1 = scandir($dir);
        $files2 = scandir($dir, 1);
        echo '<p>Upload directory is '.$dir.'</p>';
        echo '<p>Directory Listing in alphabetical order, ascending:</p><ul>';
        foreach($files1 as $file)
        {
                if ($file != "." && $file != "..")
                {
                        echo '<li>'.$file.'</li>';
                }
        }
        echo '</ul>';
```

scandir() function

```
        echo '<p>Upload directory is '.$dir.'</p>';
        echo '<p>Directory Listing in alphabetical, descending:</p><ul>';
        foreach($files2 as $file)
        {
                if ($file != "." && $file != "..")
                {
                        echo '<li>'.$file.'</li>';
                }
        }
        echo '</ul>';
?>
</body>
</html>
```

This lecture covers

- Getting data from forms
- html form + php action on the same page
  - `$_SERVER['PHP_SELF']`
  - maintain form status
- page redirection - the `header()` function
- Upload files
- **Files**
  - creation and deletion
  - opening and closing
  - reading …
- `print_r` function

Open/Create and delete files

- function `touch ()`
  - argument: string representing url of the file
  - if file did not already exist → create empty file
  - otherwise the old content is not changed but update `date modified` field
  - example
    - `touch("myfile.txt");`
- `unlink()` function
  - delete a file
  - `unlink("myfile.txt");`

Open/Create and delete files

- `fopen()` function
  - 2 arguments: file name and opening mode
  - mode: read (r), write (w), append (a)
- `fopen()` returns
  - file resource
  - or false if failed to open the file

Open/Create and delete files

- example
  - `$fp = fopen("test.txt", 'r');`

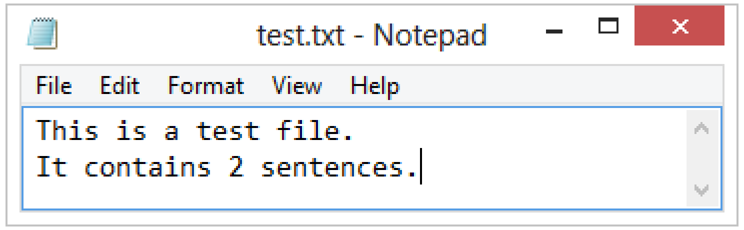
  better
  - `if ($fp = fopen("test.txt", "w")) { …}`
  - **`($fp = fopen("test.txt", "w")) or die ("… , sorry");`**
- `fclose()` function→ closes the opened file with `fopen()`
  - `fclose($fp);`

Read files line by line

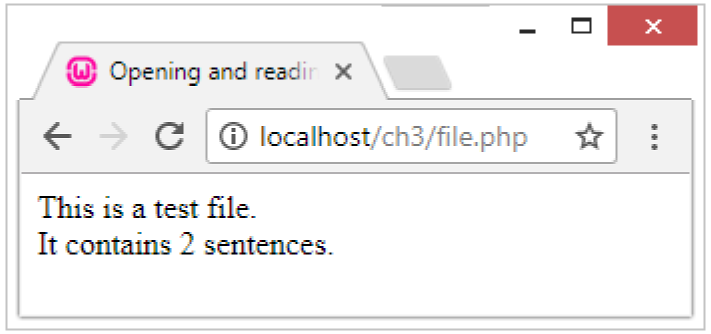
- `fgets()` function
  - ex: `$line = fgets($fp, 1024);`
  - Reads 1024 or to the end of the line \ n

- `feof()` function → return true if end of file
  - ex: `feof($fp);`

Read files line by line



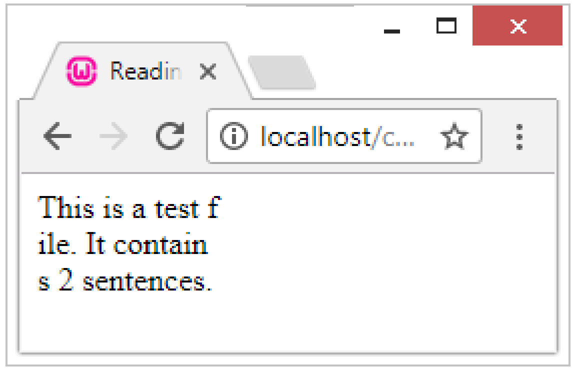
```
This is a test file.
It contains 2 sentences.
```



This is a test file.
It contains 2 sentences.

```
<html>
<head><title>Opening and reading a file line
by line</title></head>
<body>
<?php
        $filename = "test.txt";
        $fp = fopen($filename, "r") or
                die("Couldn't open $filename");
        while (!feof($fp))
        {
                $line = fgets($fp, 1024);
                print "$line<br>";
        }
?>
</body>
</html>
```

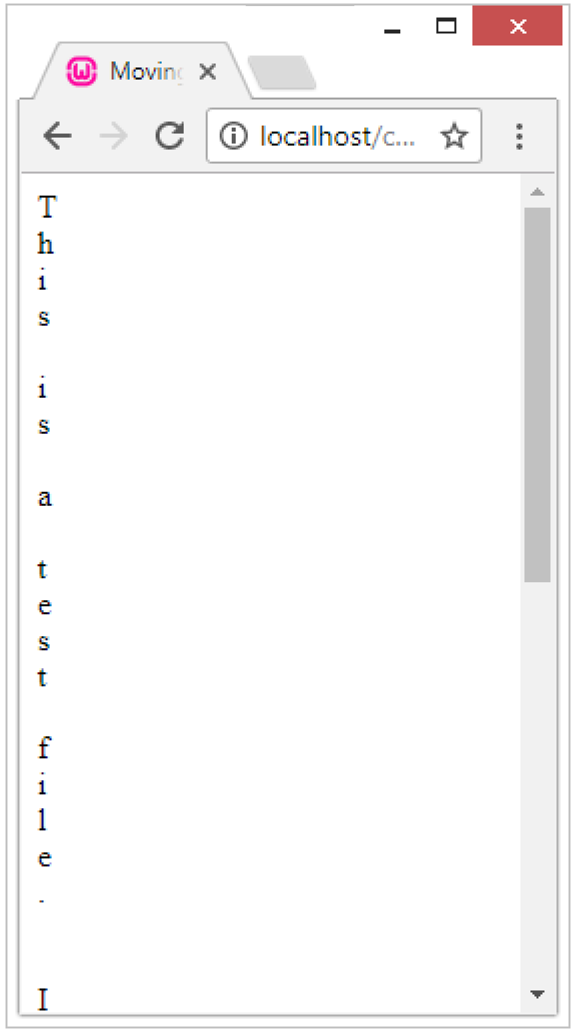
Reading Arbitrary Amounts of Data from a File with `fread()`



This is a test f
ile. It contain
s 2 sentences.

```
<html>
<head><title>Reading a file with
fread()</title></head>
<body>
<?php
        $filename = "test.txt";
        $fp = fopen($filename, "r")
                or die("Couldn't open $filename");
        while (!feof($fp))
        {
                $chunk = fread($fp, 16);
                print "$chunk<br>";
        }
?>
</body>
</html>
```

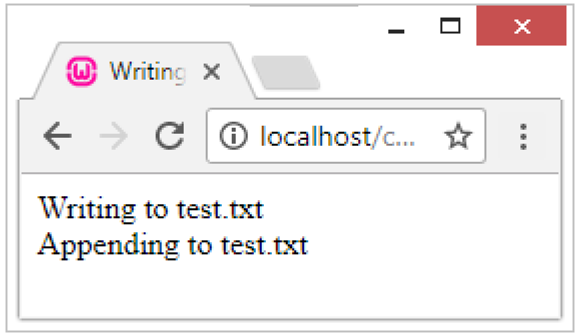
Moving Around a File with fgetc()



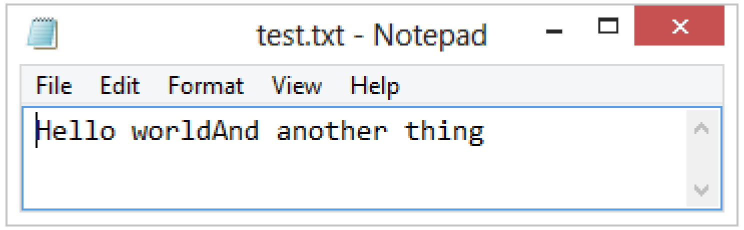
```
T
h
i
s

i
s

a

t
e
s
t

f
i
l
e
.


I
```

```
<html>
<head><title>Moving Around a File with
fseek()</title></head>
<body>
<?php
        $filename = "test.txt";
        $fp = fopen($filename, "r")
                or die("Couldn't open $filename");
        while (!feof($fp))
        {
                $char = fgetc($fp);
                print "$char<BR>";
        }
?>
</body>
</html>
```

Writing and Appending to a File



```
Writing to test.txt
Appending to test.txt
```

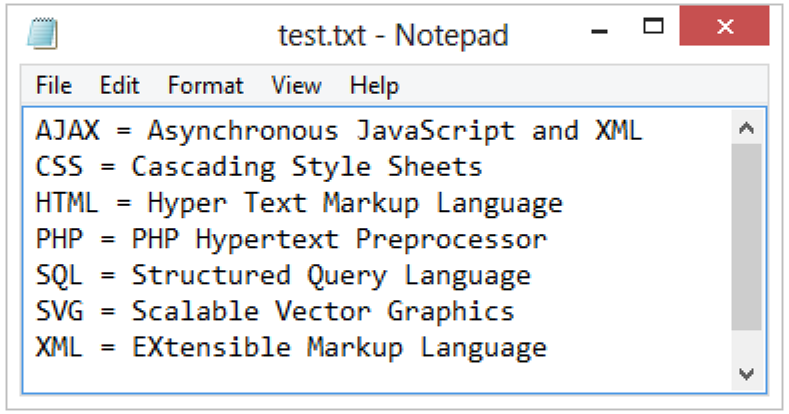


```
Hello worldAnd another thing
```

```
<html>
<head><title>Writing and appending to a
file</title></head>
<body>
<?php
        $filename = "test.txt";
        print "Writing to $filename<br>";
        $fp = fopen($filename, "w")
                or die("Couldn't open $filename");
        fwrite($fp, "Hello world\n");
        fclose($fp);

        print "Appending to $filename<br>";
        $fp = fopen($filename, "a")
                or die("Couldn't open $filename");
        fputs($fp, "And another thing\n");
        fclose($fp);
?>
</body>
</html>
```

readfile() Function

- The `readfile()` function reads a file and writes it to the output buffer.
- Assume we have a text file called “test.txt", stored on the server, that looks like this:



```
AJAX = Asynchronous JavaScript and XML
CSS = Cascading Style Sheets
HTML = Hyper Text Markup Language
PHP = PHP Hypertext Preprocessor
SQL = Structured Query Language
SVG = Scalable Vector Graphics
XML = EXtensible Markup Language
```

- The PHP code to read the file and write it to the output buffer is as follows (the `readfile()` function returns the number of bytes read on success):

`readfile()` Function

- Example

-
  ```
  <?php
        echo readfile("test.txt");
  ?>
  ```

- The `readfile()` function is useful if all you want to do is open up a file and read its contents.

`file()` function

- `file()` - Reads the file and returns the result in an array

Discover all by yourself

- `fwrite()` - Write a file in binary mode
- `fread()` - Read the file in binary mode
- `fgets()` - Retrieves the current line on which the file pointer is located
- `fgetss()` - Returns the current line of the file and removes the HTML tags
- `fscanf()` - Analyzes a file according to a format
- `fprintf()`
- `file()` - Reads the file and returns the result in a table
- `fpassthru()` - Show the rest of the file
- `rewind()` - Replace the file pointer at the beginning
- `file_get_contents()` - Get a whole file in a string
- `fseek()`
- `flock()`
- `mkdir()`
- `rmdir()`
- `opendir()`
- `readfile()` reads the filename file and sends it to the standard output

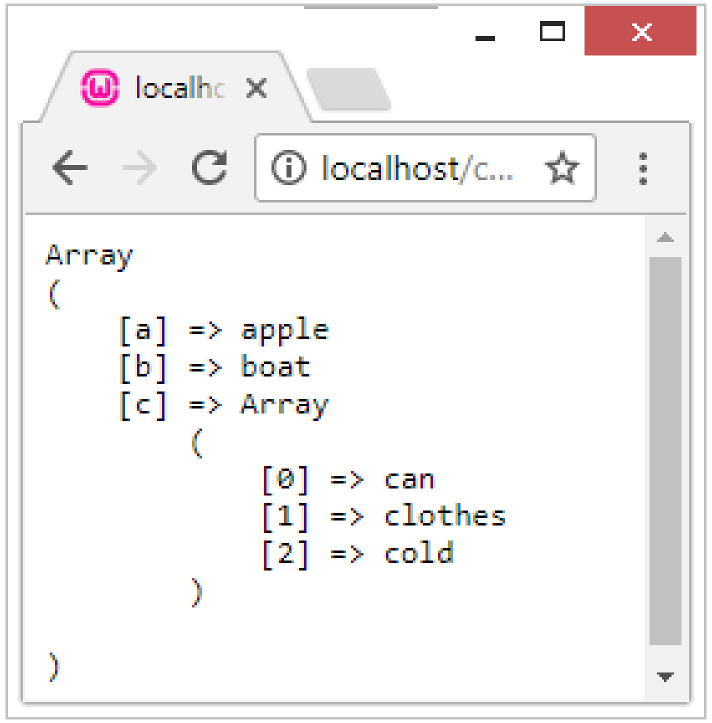
This lecture covers

- Getting data from forms
- html form + php action on the same page
  - `$_SERVER['PHP_SELF']`
  - maintain form status
- page redirection - the `header()` function
- Upload files
- Files
  - creation and deletion
  - opening and closing
  - reading …
- **`print_r` function**

`print_r` function

- `bool print_r( mixed expression [, bool return] )`
- displays information about a variable
  - so that it is legible
  - string, integer, double → value itself displayed
  - array → values presented in a format that shows the keys and values
- `print_r()` places the array pointer at the end of the array
  - `reset()` to bring it back to the beginning
- to get the result of `print_r()` in a string
  - use the return parameter → TRUE

Example with print_r()



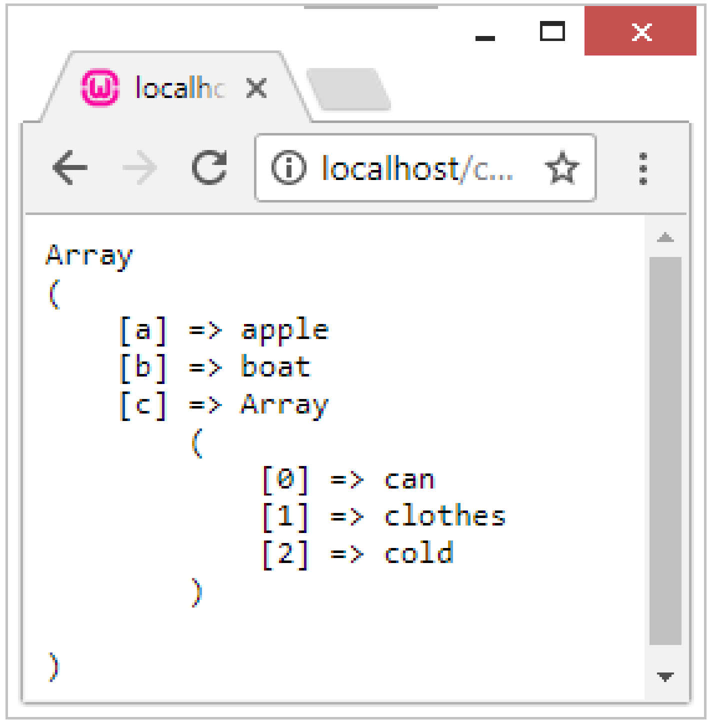
localhc

localhost/c...

```
Array
(
    [a] => apple
    [b] => boat
    [c] => Array
        (
            [0] => can
            [1] => clothes
            [2] => cold
        )

)
```

```
<pre>
<?php
        $a=array('a'=>'apple',
                 'b'=>'boat',
                 'c'=>array('can','clothes','cold'));
        print_r($a);
?>
</pre>
```

Example with print_r()



```
Array
(
    [a] => apple
    [b] => boat
    [c] => Array
        (
            [0] => can
            [1] => clothes
            [2] => cold
        )

)
```

```
<pre>
<?php
        $a=array('a'=>'apple',
                 'b'=>'boat',
                 'c'=>array('can','clothes','cold'));
        print_r($a);
        $result = print_r($a, true);

?>
</pre>
```

Attention

- `print_r()` will loop to infinity if
  - An array or an object contains a reference on itself
- classic example
  - `print_r($GLOBALS)`
  - because `$GLOBALS` is itself a global variable
  - therefore, contains a reference on itself