PHP Include Statement

PHP include statement

- The include statement takes all the text/code/markup that exists in the specified file and copies it into the file that uses the include statement.

- Including files is very useful when you want to include the same PHP, HTML, or text on multiple pages of a website.

Syntax

```
include 'filename';
```

PHP include Example

Assume we have a standard footer file called "footer.php", that looks like this:

```
<?php

echo "<p>Copyright; 1990-" . date("Y") . "
ABC.com</p>";

?>
```

PHP include Example (Continued)

To include the footer file in a page, use the include statement:

```
<html>
 <body>

<h1>Welcome to my home page!</h1>
 <p>Some text.</p>
<p>Some more text.</p>

<?php include 'footer.php';?>

</body>
 </html>
```

PHP Date and Time

PHP Date and Time

- The PHP date() function is used to format a date and/or a time.

The PHP Date() Function

- The PHP date() function formats a timestamp to a more readable date and time.

Syntax

date(format,timestamp)

Parameter	Description
format	Required. Specifies the format of the timestamp
timestamp	Optional. Specifies a timestamp. Default is the current date and time

Note that the PHP date() function will return the current date/time of the server!

Get a Simple Date

- The required format parameter of the date() function specifies how to format the date (or time).

- Here are some characters that are commonly used for dates:
  - d – Represents the day of the month (01 to 31)
  - m – Represents a month (01 to 12)
  - Y – Represents a year (in four digits)
  - l (lowercase 'L') – Represents the day of the week

- Other characters, like"/", ".", or "-" can also be inserted between the characters to add additional formatting.

Example

- The example below formats today's date in three different ways:

```
<?php
echo "Today is " . date("Y/m/d") . "<br>";
echo "Today is " . date("Y.m.d") . "<br>";
echo "Today is " . date("Y-m-d") . "<br>";
 echo "Today is " . date("l");
?>
```

Automatic Copyright Year

- Use the date() function to automatically update the copyright year on your website:

Example

```
&copy; 2010-<?php echo date("Y")?>
```

miss ;

Get a Simple Time

- Here are some characters that are commonly used for times:
  - h – 12-hour format of an hour with leading zeros (01 to 12)
  - i – Minutes with leading zeros (00 to 59)
  - s – Seconds with leading zeros (00 to 59)
  - a – Lowercase Ante meridiem and Post meridiem (am or pm)

Example

- The example below outputs the current time in the specified format:

```
<?php
echo "The time is " . date("h:i:sa");
?>
```

Get Your Time Zone

- If the time you got back from the code is not the right time, it's probably because your server is in another country or set up for a different timezone.

- So, if you need the time to be correct according to a specific location, you can set a timezone to use.

Get Your Time Zone (Example)

- The example below sets the timezone to "Asia/Karachi", then outputs the current time in the specified format:

Example

```
<?php
date_default_timezone_set("Asia/Karachi");
echo "The time is " . date("h:i:sa");
?>
```

PHP Cookies

What is a Cookie?

- A cookie is often used to identify a user. A cookie is a small file that the server embeds on the user's computer.
- Each time the same computer requests a page with a browser, it will send the cookie too.
- With PHP, you can both create and retrieve cookie values.

Create Cookies With PHP

A cookie is created with the setcookie() function.

Syntax

```
setcookie(name, value, expire, path, domain, secure, httponly);
```

Only the name parameter is required. All other parameters are optional.

Note: The setcookie() function must appear BEFORE the <html> tag.

Example

Modify= "Ahmed Cookie"

```
<?php
$cookie_name = "user";
$cookie_value = "Ahmed";
setcookie($cookie_name, $cookie_value, time() + (86400
* 30), "/"); // 86400 = 1 day
?>
<html><body>
<?php
if(!isset($_COOKIE[$cookie_name])) {
    echo "Cookie named '" . $cookie_name . "' is not set!";
} else {
    echo "Cookie '" . $cookie_name . "' is set!<br>";
    echo "Value is: " . $_COOKIE[$cookie_name];}
?>
</body></html>
```

F5 return < Old

The value of the cookie is automatically URL encoded when sending the cookie, and automatically decoded when received (to prevent URL encoding, use setrawcookie() instead).

Delete a Cookie

To delete a cookie, use the setcookie() function with an expiration date in the past:

Example

```
<?php
// set the expiration date to one hour ago
setcookie("user", "", time() - 3600,"/");
?>
<html><body>
<?php
echo "Cookie 'user' is deleted.";
?>
</body></html>
```

ENABLE

PHP Sessions

What is a Session?

- A session is a way to store information (in variables) to be used across several pages.
- Unlike a cookie, the information is not stored on the users computer.
- When you work with an application, you open it, do some changes, and then you close it. This is much like a Session.
- On the internet there is one problem: the web server does not know who you are, because the HTTP address doesn't maintain state.
- Session variables solve this problem by storing user information to be used across multiple pages (e.g. username etc.). By default, session variables last until the user closes the browser.
- Session variables hold information about one single user, and are available to all pages in one application.

Start a PHP Session

- A session is started with the session_start() function.
- Session variables are set with the PHP global variable: $_SESSION.
- Now, let's create a new page called "session_demo1.php". In this page, we start a new PHP session and set some session variables:

Example

```
<?php
// Start the session
session_start();
?>
<!DOCTYPE html>
<html><body>
<?php
// Set session variables
$_SESSION["favcolor"] = "green";
$_SESSION["favanimal"] = "cat";
echo "Session variables are set.";
?>
</body></html>
```

Array

Note: The session_start() function must be the very first thing in your document. Before any HTML tags.

Get PHP Session Variable Values

- Next, we create another page called "session_demo2.php". From this page, we will access the session information we set on the first page ("session_demo1.php").
- Session variables are not passed individually to each new page, instead they are retrieved from the session we open at the beginning of each page (session_start()).
- All session variable values are stored in the global $_SESSION variable

Example

Modify

```
<?php
session_start();
?>
<!DOCTYPE html>
<html>
 <body>

<?php
// Echo session variables that were set on previous page
echo "Favorite color is " . $_SESSION["favcolor"] . ".<br>";
echo "Favorite animal is " . $_SESSION["favanimal"] . ".";
?>

 </body>
</html>
```

Example 2

- Another way to show all the session variable values for a user session is to run the following code:

```
<?php
session_start();
?>
<!DOCTYPE html>
<html>
<body>

<?php
 print_r($_SESSION);
?>

</body>
</html>
```

How does it work?

- Most sessions set a user-key on the user's computer that looks something like this: 765487cf34ert8dede5a562e4f3a7e12.
- Then, when a session is opened on another page, it scans the computer for a user-key.
- If there is a match, it accesses that session, if not, it starts a new session.

How does it work?

- Most sessions set a user-key on the user's computer that looks something like this: 765487cf34ert8dede5a562e4f3a7e12.
- Then, when a session is opened on another page, it scans the computer for a user-key.
- If there is a match, it accesses that session, if not, it starts a new session.

Destroy a PHP Session

To remove all global session variables and destroy the session, use session_unset() and session_destroy():

Example

```
<?php
session_start();
?>
<!DOCTYPE html>
<html><body>

<?php
// remove all session variables
session_unset();

// destroy the session
session_destroy();
?>

</body></html>
```

PHP Filters

Why Use Filters?

Filters are used for

- Validating data = Determine if the data is in proper form.
- Sanitizing data = Remove any illegal character from the data.
- Many web applications receive external input. External input/data can be:
  - User input from a form
  - Cookies
  - Web services data
  - Server variables
  - Database query results

VIP

- You should always validate external data!
- Invalid submitted data can lead to security problems and break your webpage!
- By using PHP filters you can be sure your application gets the correct input!

PHP filter_var() Function

- The filter_var() function both validate and sanitize data.

- The filter_var() function filters a single variable with a specified filter. It takes two pieces of data:
  - The variable you want to check
  - The type of check to use

Sanitize a String

- The following example uses the filter_var() function to remove all HTML tags from a string:

Example

```
<?php
$str = "<h1>Hello World!</h1>";
$newstr = filter_var($str,
FILTER_SANITIZE_STRING);
echo $newstr;
?>
```

Sanitize and Validate a URL

- The following example uses the filter_var() function to first remove all illegal characters from a URL, then check if $url is a valid URL:

Example

```
<?php
$url = "http://www.vu.edu.pk";

// Remove all illegal characters from a url
$url = filter_var($url, FILTER_SANITIZE_URL);

// Validate url
if (!filter_var($url, FILTER_VALIDATE_URL) === false) {
    echo("$url is a valid URL");
} else {
    echo("$url is not a valid URL");
}
?>
```

Sanitize and Validate an Email Address

- The following example uses the filter_var() function to first remove all illegal characters from the $email variable, then check if it is a valid email address:

Example

```
<?php
$email = "john.doe@example.com";

// Remove all illegal characters from email
$email = filter_var($email, FILTER_SANITIZE_EMAIL);

// Validate e-mail
if (!filter_var($email, FILTER_VALIDATE_EMAIL) === false) {
    echo("$email is a valid email address");
} else {
    echo("$email is not a valid email address");
}
?>
```

Validate an Integer

- The following example uses the filter_var() function to check if the variable $int is an integer. If $int is an integer, the output of the code above will be: "Integer is valid". If $int is not an integer, the output will be: "Integer is not valid":

Example

```
<?php
$int = 100;

if (!filter_var($int, FILTER_VALIDATE_INT) === false) {
    echo("Integer is valid");
} else {
    echo("Integer is not valid");
}
?>
```

identical

Tip: filter_var() and Problem With 0

- In the previous example, if $int was set to 0, the function above will return "Integer is not valid". To solve this problem, use the code below:

Example

```
<?php
$int = 0;

if (filter_var($int, FILTER_VALIDATE_INT) === 0 || !filter_var($int,
FILTER_VALIDATE_INT) === false) {
    echo("Integer is valid");
} else {
    echo("Integer is not valid");
}
?>
```

Validate an IP Address

- The following example uses the filter_var() function to check if the variable $ip is a valid IP address:

Example

```
<?php
$ip = "127.0.0.1";

if (!filter_var($ip, FILTER_VALIDATE_IP) === false) {
    echo("$ip is a valid IP address");
} else {
    echo("$ip is not a valid IP address");
}
?>
```

Introduction to MySQL Database

Introduction to Databases

- **Data base**
  - A database is an organized collection of data.
- **Database management system (DBMS)**
  - It is a computer software application, designed to allow the definition, creation, querying, update, and administration of databases.
  - **Example**

    Some examples of DBMS are :
    - MySQL
    - PostgreSQL
    - Microsoft SQL Server
    - Oracle
    - Sybase

MySQL Database

- A free, fast, reliable, easy-to-use, multi-user multi-threaded relational database system.
- SQL (Structured Query Language) is use in MYSQL database system.
- It is freely available and released under GPL (GNU General Public License).
- Officially pronounced “my Ess Que Ell” (not my sequel).

Use of Database tools

Database tools

- There are many tools available for handling MY SQL databases such as:

- **XAMPP** :consisting mainly of the Apache HTTP Server, MySQL database
- **WAMP**:consisting of the Apache web server, OpenSSL for SSL support, MySQL database

Secure Sockets Layer

- For our course we will use **WAMP**

PhpMyAdmin

- Is a FREE software tool written in PHP intended to handle the administration of MySQL over the Internet.
- PhpMyAdmin supports a wide range of operations with MySQL, the most frequently used being the managing of databases, tables, fields, relations, indexes, users, permissions.
- You still have the ability to execute SQL statements directly as well.

How to use MY SQL with XAMPP server

Privileges and Security

Privileges and Security

- These mysql database tables contain grant information:
- **user:** User accounts, global privileges, and other non-privilege columns.
- **db:** Database-level privileges.
- **host:** Obsolete.
- **tables_priv:** Table-level privileges.
- **columns_priv:** Column-level privileges.
- **procs_priv**: Stored procedure and function privileges.

Privileges and Security

- Used in PHP code,
- <?php

```
$servername = "localhost";
$username = "username";
$password = "password";

// Create connection
$conn = mysqli_connect($servername, $username,
$password);

// Check connection
if (!$conn) {
else   die("Connection failed: " . mysqli_connect_error());
}
echo "Connected successfully";
?>
```

MySQL Connect

MySQL Connect

- Earlier versions of PHP used the MySQL extension. PHP version 5 and after that work with a MySQL database using:

- **MySQLi extension** (the "i" stands for improved)
- **PDO (PHP Data Objects)**

MySQL Connect

- PDO will work on 12 different database systems, where as MySQLi will only work with MySQL databases.
- Three ways of working with PHP and MySQL:

  i- MySQLi (object-oriented)

  ii- MySQLi (procedural)

  iii- PDO

Note: We will use Procedural method in our examples.

MySQL Create DB

MySQL Create DB

The CREATE DATABASE statement is used to create a database in MySQL.

Examples

```
<?PHP   $servername = "localhost";

$username = “root";

$password = “";
```

MySQL Create DB (example continued)

```
// Create connection
$conn = mysqli_connect($servername, $username, $password);

// Check connection
if (!$conn) {
   die("Connection failed: " . mysqli_connect_error());}


      // Create database
$sql = "CREATE DATABASE mystudent";
if (mysqli_query($conn, $sql)) {
   echo "Database created successfully";
} else {
   echo "Error creating database: " . mysqli_error($conn);
}   ?>
```

MySQL Create Table

MySQL Create Table

- The CREATE TABLE statement is used to create a table in MySQL.
- We will create a table named “Student", with five columns: “s_id", "firstname", "lastname", "email" and "reg_date":

SQL Statement:-

```
CREATE TABLE Student (
s_id INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
firstname VARCHAR(30) NOT NULL,
lastname VARCHAR(30) NOT NULL,
email VARCHAR(50),
reg_date TIMESTAMP
)
```

MYSQL Create Table

Example:-

```
<?php
$servername = "localhost";
$username = “root";
$password = “";
$dbname = "mystudent";

// Create connection
$conn = mysqli_connect($servername, $username,
$password, $dbname);
// Check connection
if (!$conn) {
   die("Connection failed: " . mysqli_connect_error());
}
```

MySQL Create Table (Example Continued)

```
// sql to create table
$sql = "CREATE TABLE student (
id INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
firstname VARCHAR(30) NOT NULL,
lastname VARCHAR(30) NOT NULL,
email VARCHAR(50),
reg_date TIMESTAMP
)";

if (mysqli_query($conn, $sql)) {
    echo "Table student created successfully";
} else {
    echo "Error creating table: " . mysqli_error($conn);
}

mysqli_close($conn);
?>
```

MySQL Insert Data

MySQL Insert Data

- When a database and a table have been created, we can start adding data in them.

Here are some syntax rules to follow:

- The SQL query must be quoted in PHP
- String values inside the SQL query must be quoted
- Numeric values must not be quoted
- The word NULL must not be quoted
- The INSERT INTO statement is used to add new records to a MySQL table

MySQL Insert Data

- The INSERT INTO statement is used to add new records to a MySQL table:

SQL Statement: INSERT INTO table_name (column1, column2, column3,...) VALUES (value1, value2, value3,...)

CV

Note: If a column is AUTO_INCREMENT (like the "id" column) or TIMESTAMP (like the "reg_date" column), it is no need to be specified in the SQL query; MySQL will automatically add the value.

MySQL Insert Data

Example

```
<?php
$servername = "localhost";
$username = “root";
$password = “";
$dbname = "mystudent";

// Create connection
$conn = mysqli_connect($servername, $username,
$password, $dbname);
// Check connection
if (!$conn) {
   die("Connection failed: " . mysqli_connect_error());
}
```

MySQL Insert Data

Example(continued)

```
$sql = "INSERT INTO student (firstname, lastname, email)
VALUES (‘Ali', ‘Wali', ‘ali@example.com')";

if (mysqli_query($conn, $sql)) {
    echo "New record created successfully";
} else {
    echo "Error: " . $sql . "<br>" . mysqli_error($conn);
}

mysqli_close($conn);
?>
```

MySQL Get Last ID

MySQL Get Last ID

- we perform an INSERT or UPDATE on a table with an AUTO_INCREMENT field, we can get the ID of the last inserted/updated record immediately.
- In the table “student", the "id" column is an AUTO_INCREMENT field:

Example

```
<?php
$servername = "localhost";
$username = “root";
$password = “";
$dbname = "mystudent";
```

MySQL Get Last ID

```
// Create connection
$conn = mysqli_connect($servername, $username, $password,
$dbname);
// Check connection
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}

$sql = "INSERT INTO student (firstname, lastname, email)
VALUES ('Nadeem', 'Muhammad', 'Nadeem@example.com')";

if (mysqli_query($conn, $sql)) {
    $last_id = mysqli_insert_id($conn);
    echo "New record created successfully. Last inserted ID is: " .
$last_id;
} else {
    echo "Error: " . $sql . "<br>" . mysqli_error($conn);
}

mysqli_close($conn);
?>
```

MySQL Insert in Multiple.

MySQL Insert in Multiple.

- Multiple SQL statements must be executed with the mysqli_multi_query() function.

- The following examples add three new records to the “student" table:

```
<?php
$servername = "localhost";
$username = “root";
$password = “";
$dbname = "mystudent";
```

MySQL Insert in Multiple.

```
$conn = mysqli_connect($servername, $username,
$password, $dbname);
// Check connection
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}

$sql = "INSERT INTO student (firstname, lastname, email)
VALUES (‘Babar', ‘Ali', ‘Babar@example.com');";
$sql .= "INSERT INTO student (firstname, lastname, email)
VALUES (‘sara', ‘khan', ‘sara@example.com');";
$sql .= "INSERT INTO  student (firstname, lastname,
email)
VALUES (‘Mahmood', ‘Zakir', ‘Mahmood@example.com')";
```

MySQL Insert in Multiple.

```
if (mysqli_multi_query($conn, $sql)) {
    echo "New records created successfully";
} else {
    echo "Error: " . $sql . "<br>" .
mysqli_error($conn);
}

mysqli_close($conn);
?>
```

MySQL Prepared.

MySQL Prepared.

- A prepared statement is a feature used to execute the same (or similar) SQL statements repeatedly with high efficiency.
- Prepared statements basically work like this:
- Prepare: An SQL statement template is created and sent to the database. Certain values are left unspecified, called parameters (labeled "?"). Example: INSERT INTO student VALUES(?, ?, ?)
- The database parses, compiles, and performs query optimization on the SQL statement template, and stores the result without executing it
- Execute: At a later time, the application binds the values to the parameters, and the database executes the statement.

MySQL Prepared.

- The application may execute the statement as many times as it wants with different values
- Compared to executing SQL statements directly, prepared statements have two main advantages:

- Prepared statements reduces parsing time as the preparation on the query is done only once (although the statement is executed multiple times)
- Bound parameters minimize bandwidth to the server as you need send only the parameters each time, and not the whole query
- Prepared statements are very useful against SQL injections, because parameter values, which are transmitted later using a different protocol, need not be correctly escaped. If the original statement template is not derived from external input, SQL injection cannot occur.

MySQL Prepared.

Example

```
<?php
$servername = "localhost";
$username = “root";
$password = "";
$dbname = "mystudent";

// Create connection
$conn = new mysqli($servername, $username, $password,
$dbname);

// Check connection
if ($conn->connect_error) {
   die("Connection failed: " . $conn->connect_error);
}
```

MySQL Prepared.

```
// prepare and bind
$stmt = $conn->prepare("INSERT INTO student
(firstname, lastname, email) VALUES (?, ?, ?)");
$stmt->bind_param("sss", $firstname, $lastname,
$email);

// set parameters and execute
$firstname = “Khalid";
$lastname = “Zia";
$email = “khakid_zia@example.com";
$stmt->execute();
```

MySQL Prepared.

```
$firstname = “Amna";
$lastname = “khan";
$email = “amna@example.com";
$stmt->execute();

$firstname = “Zanub";
$lastname = “ali";
$email = “zanub@example.com";
$stmt->execute();

echo "New records created successfully";

$stmt->close();
$conn->close();
?>
```

S F
D FW
U SW

MySQL Select Data

MySQL Select Data F

- The SELECT statement is used to select data from one or more tables:

Example

SELECT column_name(s) FROM table_name

we can use the * character to select ALL columns from a table:

Example

SELECT * FROM table_name

MySQL Select Data

Example

```
<?php
$servername = "localhost";
$username = “root";
$password = "";
$dbname = "mystudent";

// Create connection
$conn = mysqli_connect($servername, $username,
$password, $dbname);
// Check connection
if (!$conn) {
   die("Connection failed: " . mysqli_connect_error());
}
```

MySQL Select Data

```
$sql = "SELECT id, firstname, lastname FROM student";
$result = mysqli_query($conn,  $sql);

if (mysqli_num_rows($result)  > 0) {
    // output data of each row
    while($row = mysqli_fetch_assoc($result)) {
        echo "id: " . $row["id"]. " - Name: " .
$row["firstname"]. " " . $row["lastname"]. "<br>";
    }
} else {
    echo "0 results";
}

mysqli_close($conn);
?>
```

OT:SS

MySQL Delete Data

MySQL Delete Data FW

The DELETE statement is used to delete records from a table.

SQL Statement:-DELETE FROM table_name WHERE some_column = some_value.

Example:-

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";
```

MySQL Delete Data

Example (continued)

```
// Create connection
$conn = mysqli_connect($servername, $username, $password,
$dbname);
// Check connection
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}

// sql to delete a record
$sql = "DELETE FROM student WHERE id=3";

if (mysqli_query($conn, $sql)) {
    echo "Record deleted successfully";
} else {
    echo "Error deleting record: " . mysqli_error($conn);
}
mysqli_close($conn);
?>
```

MySQL Update Data

MySQL Update Data

- The UPDATE statement is used to update existing records in a table:
- SQL UPDATE Statement:- UPDATE table_name SET column1=value, column2=value2,... WHERE some_column=some_value

Example:-

```
<?php
$servername = "localhost";
$username = “root";
$password = "";
$dbname = "mystudent";
```

MySQL Update Data

SW

Example(continued)

```
// Create connection
$conn = mysqli_connect($servername, $username, $password,
$dbname);
// Check connection
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}

$sql = "UPDATE student SET lastname='Ismaile'  WHERE id=2";

if (mysqli_query($conn, $sql)) {
    echo "Record updated successfully";
} else {
    echo "Error updating record: " . mysqli_error($conn);
}

mysqli_close($conn);
?>
```

MySQL Limit Data

MySQL Limit Data

- MySQL provides a LIMIT clause that is used to specify the number of records to return

Example:-

$sql = "SELECT * FROM student Orders LIMIT 05";

When the SQL query above is run, it will return the first 05 records.

PHP File Handling

PHP File Handling

- File handling is an important part of any web application. You often need to open and process a file for different tasks.

PHP Manipulating Files

- PHP has several functions for creating, reading, uploading, and editing files.

- **Be careful when manipulating files!**

  When you are manipulating files you must be very careful. You can do a lot of damage if you do something wrong.

  - **Common errors are**:

    - Editing the wrong file
    - Filling a hard-drive with garbage data
    - And deleting the content of a file by accident

PHP readfile() Function

PHP File Handling

PHP readfile() Function

- The readfile() function reads a file and writes it to the output buffer.

- Assume we have a text file called "**webdictionary.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

PHP readfile() Function (Cont.)

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):

Example

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

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

Web Browser output

192.168.1.102:8080/dcs/Fi

192.168.1.102:8080/dcs/Files/1-readfile.php

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 238

PHP Open File – fopen()

PHP File Handling

PHP Open File – fopen()

A better method to open files is with the fopen() function. This function gives you more options than the readfile() function.

We will use the text file, "**webdictionary.txt**", during the lessons:

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

fopen() function

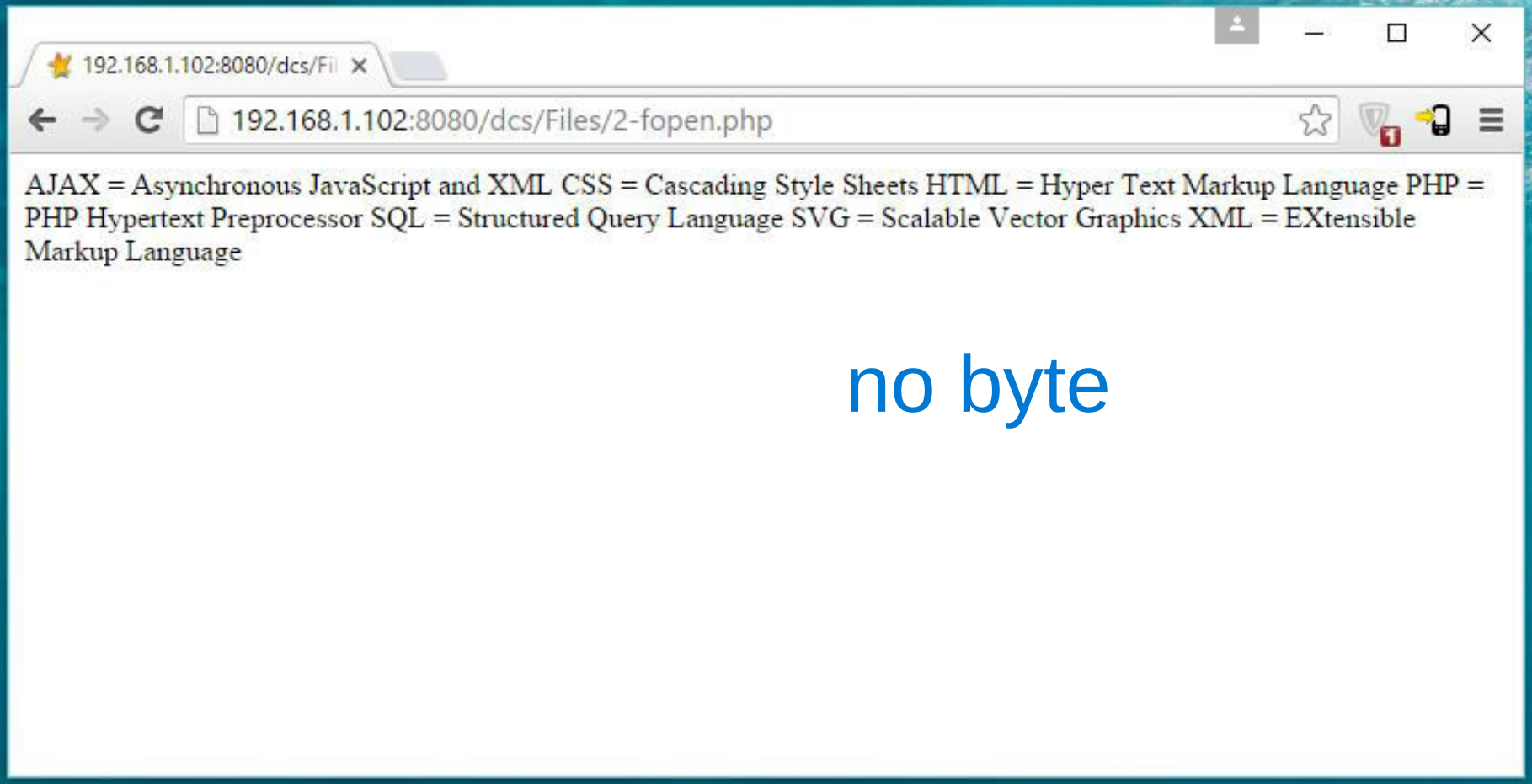
- fopen() function looks like this
  fopen("webdictionary.txt", "r")

- The first parameter of fopen() contains the name of the file to be opened and the second parameter specifies in which mode the file should be opened.

Example

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

Web Browser Output



no byte

Modes to open a file

The file may be opened in one of the following modes:

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 save
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

PHP Read File – fread()

PHP File Handling

PHP Read File – fread()

The fread() function reads from an open file.

The following PHP code reads the "**webdictionary.txt**" file to the end:

```
fread($myfile,filesize("webdictionary.txt"));
```

The first parameter of fread() contains the name of the file to read from and the second parameter specifies the maximum number of bytes to read.

PHP Close File – fclose()

PHP File Handling

PHP Close File – fclose()

- The fclose() function is used to close an open file.
- It's a good programming practice to close all files after you have finished with them. You don't want an open file running around on your server taking up resources!
- The fclose() requires the name of the file (or a variable that holds the filename) we want to close:

Example

```
<?php
 $myfile = fopen("webdictionary.txt", "r");
// some code to be executed....
 fclose($myfile);
 ?>
```

PHP Read Single Line – fgets()

PHP File Handling

PHP Read Single Line – fgets()

- The fgets() function is used to read a single line from a file.

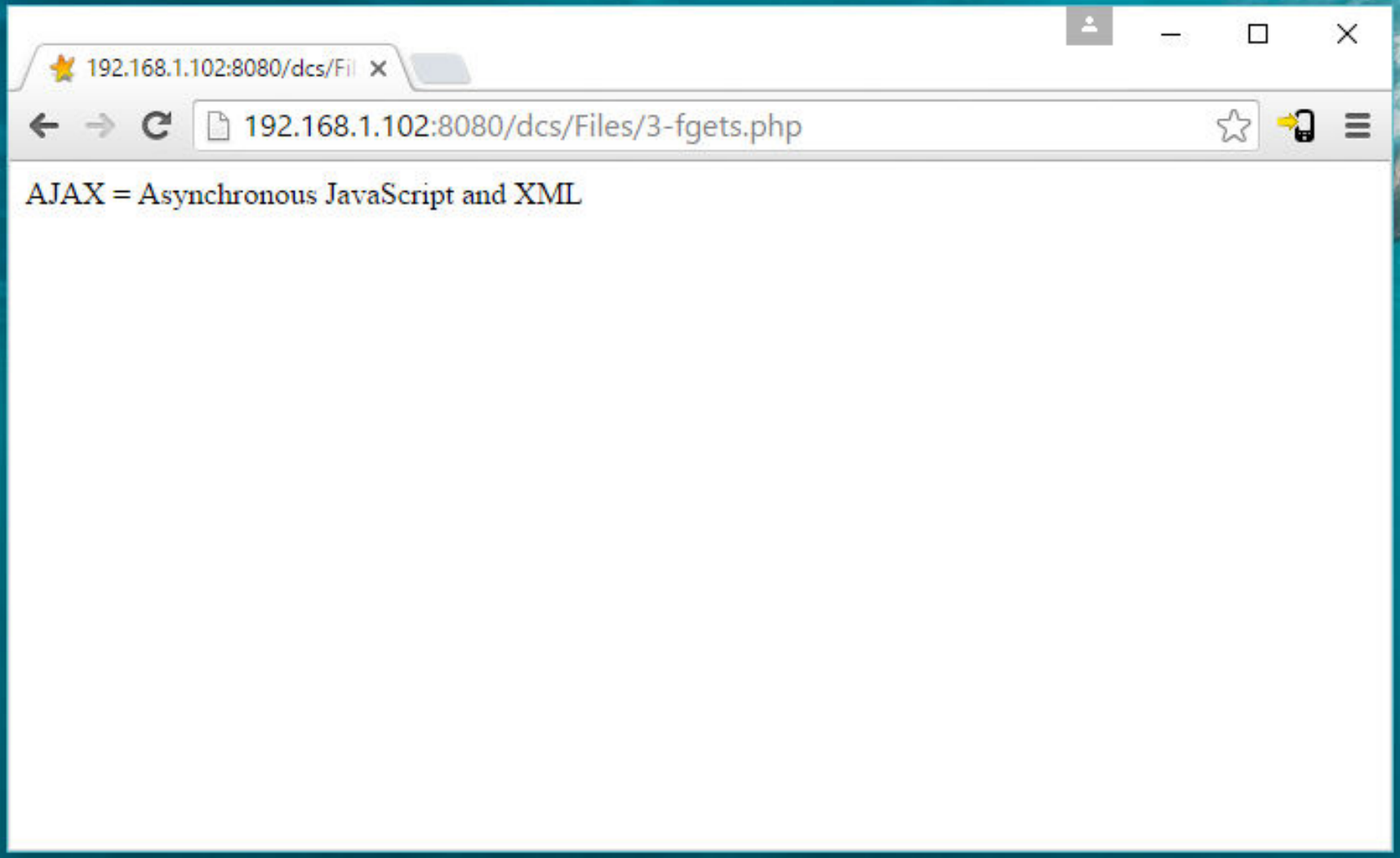
- The example below outputs the first line of the "**webdictionary.txt**" file:

Example

```
<?php
$myfile = fopen("webdictionary.txt", "r") or die("Unable to open file!");
echo fgets($myfile);
 fclose($myfile);
?>
```

Note: After a call to the fgets() function, the file pointer has moved to the next line.

Web Browser Output



PHP Check End-Of-File – feof()

PHP File Handling

PHP Check End-Of-File - feof()

- The feof() function checks if the "end-of-file" (EOF) has been reached.
- The feof() function is useful for looping through data of unknown length.
- The example below reads the "**webdictionary.txt**" file line by line, until end-of-file is reached:

Example

```
<?php
$myfile = fopen("webdictionary.txt", "r") or die("Unable to open file!");
// Output one line until end-of-file
while(!feof($myfile)) {
  echo fgets($myfile) . "<br>";
}
fclose($myfile);
?>
```

Web Browser Output

192.168.1.102:8080/dcs/Fi

192.168.1.102:8080/dcs/Files/4-feof.php

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

again revise video start

PHP Read Single Character – fgetc()

PHP File Handling

PHP Read Single Character – fgetc()

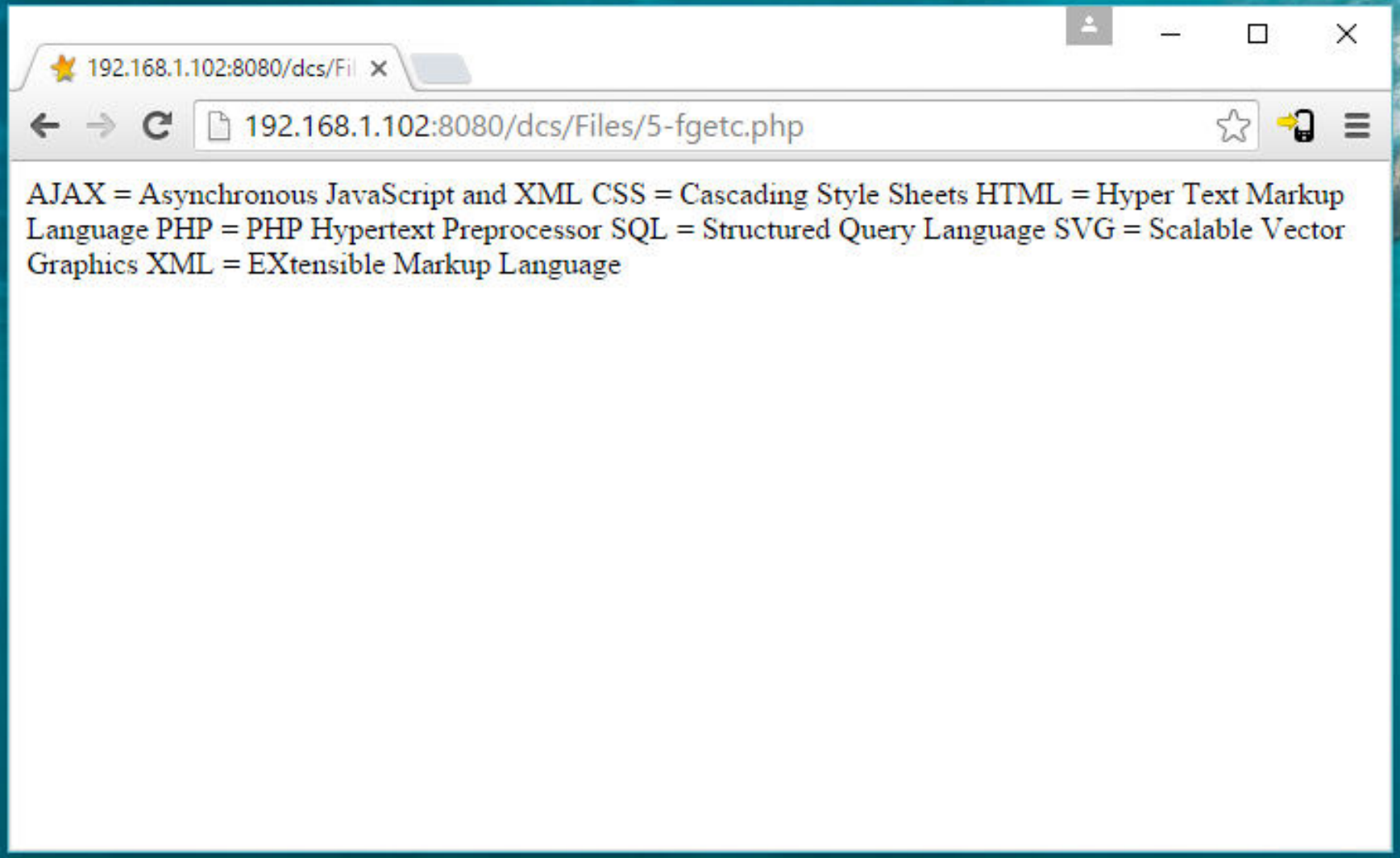
- The fgetc() function is used to read a single character from a file.
- The example below reads the "webdictionary.txt" file character by character, until end-of-file is reached:

Example

```
<?php
$myfile = fopen("webdictionary.txt", "r") or die("Unable to open file!");
// Output one character until end-of-file
while(!feof($myfile)) {
  echo fgetc($myfile);
}
fclose($myfile);
?>
```

Note: After a call to the fgetc() function, the file pointer moves to the next character.

Web Browser Output



Creating a Custom Exception Class

PHP Exception Handling

Creating a Custom Exception Class

- Creating a custom exception handler is quite simple.
- We simply create a special class with functions that can be called when an exception occurs in PHP.
- The class must be an extension of the exception class.
- The custom exception class inherits the properties from PHP's exception class and you can add custom functions to it.

Example

```
<?php
 class customException extends Exception {
   public function errorMessage() {
     //error message
     $errorMsg = 'Error on line '.$this->getLine().' in '.$this->getFile()
     .': <b>'.$this->getMessage().'</b> is not a valid E-Mail address';
     return $errorMsg;
   } }
$email = "someone@example...com";
 try {
   //check if
   if(filter_var($email, FILTER_VALIDATE_EMAIL) === FALSE) {
     //throw exception if email is not valid
     throw new customException($email);
   }
 }
catch (customException $e) {
   //display custom message
   echo $e->errorMessage(); } ?>
```

The new class is a copy of the old exception class with an addition of the errorMessage() function.
Since it is a copy of the old class, and it inherits the properties and methods from the old class, we can use the exception class methods like getLine() and getFile() and getMessage().

Example explained:

```
<?php
 class customException extends Exception {
   public function errorMessage() {
     //error message
     $errorMsg = 'Error on line '.$this->getLine().' in '.$this->getFile()
     .': <b>'.$this->getMessage().'</b> is not a valid E-Mail address';
     return $errorMsg;
   } }
$email = "someone@example...com";
 try {
   //check if
   if(filter_var($email, FILTER_VALIDATE_EMAIL) === FALSE) {
     //throw exception if email is not valid
     throw new customException($email);
   }
 }
catch (customException $e) {
   //display custom message
   echo $e->errorMessage(); } ?>
```

The **customException()** class is created as an extension of the old exception class. This way it inherits all methods and properties from the old exception class

Example explained:

```
<?php
 class customException extends Exception {
   public function errorMessage() {
     //error message
     $errorMsg = 'Error on line '.$this->getLine().' in '.$this->getFile()
     .': <b>'.$this->getMessage().'</b> is not a valid E-Mail address';
     return $errorMsg;
   } }
$email = "someone@example...com";
 try {
   //check if
   if(filter_var($email, FILTER_VALIDATE_EMAIL) === FALSE) {
     //throw exception if email is not valid
     throw new customException($email);
   }
 }
catch (customException $e) {
   //display custom message
   echo $e->errorMessage(); } ?>
```

The **errorMessage()** function is created. This function returns an error message if an e-mail address is invalid

Example explained:

```
<?php
 class customException extends Exception {
   public function errorMessage() {
     //error message
     $errorMsg = 'Error on line '.$this->getLine().' in '.$this->getFile()
     .': <b>'.$this->getMessage().'</b> is not a valid E-Mail address';
     return $errorMsg;
   } }
$email = "someone@example...com";
 try {
   //check if
   if(filter_var($email, FILTER_VALIDATE_EMAIL) === FALSE) {
     //throw exception if email is not valid
     throw new customException($email);
   }
 }
catch (customException $e) {
   //display custom message
   echo $e->errorMessage(); } ?>
```

The **$email** variable is set to a string that is not a valid e-mail address

Example explained:

```
<?php
 class customException extends Exception {
   public function errorMessage() {
     //error message
     $errorMsg = 'Error on line '.$this->getLine().' in '.$this->getFile()
     .': <b>'.$this->getMessage().'</b> is not a valid E-Mail address';
     return $errorMsg;
   } }
$email = "someone@example...com";
 try {
   //check if
   if(filter_var($email, FILTER_VALIDATE_EMAIL) === FALSE) {
     //throw exception if email is not valid
     throw new customException($email);
   }
 }
catch (customException $e) {
   //display custom message
   echo $e->errorMessage(); } ?>
```

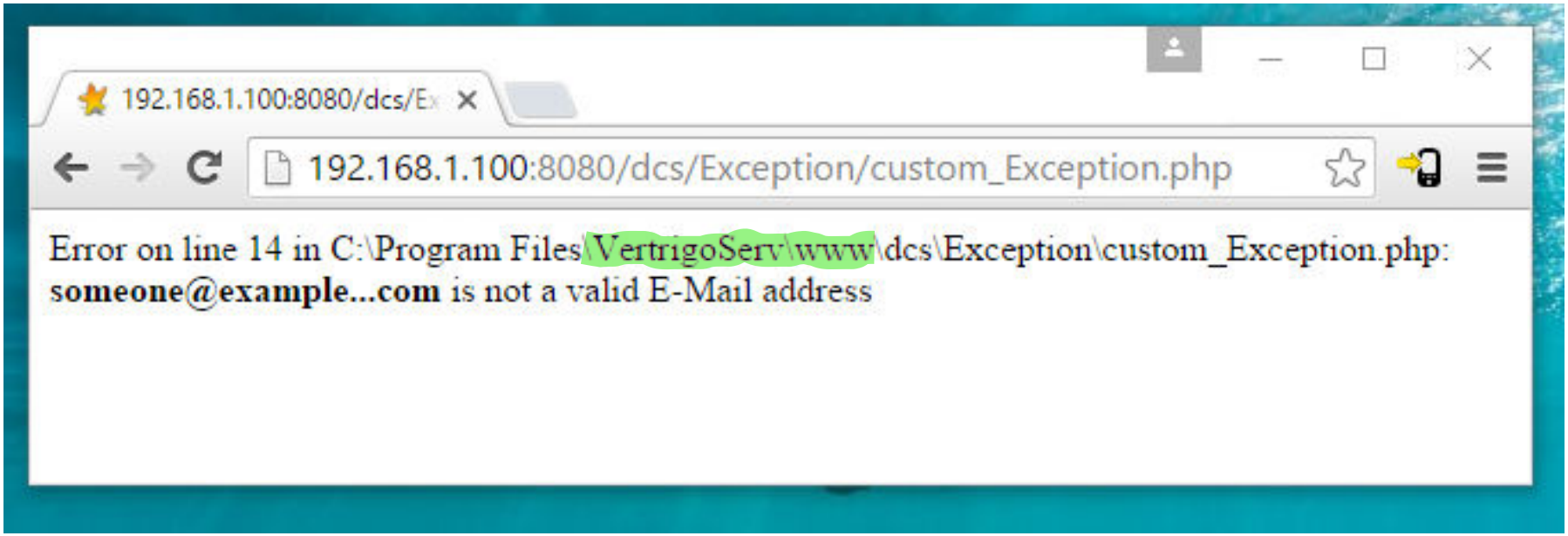
The "try" block is executed and an exception is thrown since the e-mail address is invalid

Example explained:

```
<?php
 class customException extends Exception {
   public function errorMessage() {
     //error message
     $errorMsg = 'Error on line '.$this->getLine().' in '.$this->getFile()
     .': <b>'.$this->getMessage().'</b> is not a valid E-Mail address';
     return $errorMsg;
   } }
$email = "someone@example...com";
 try {
   //check if
   if(filter_var($email, FILTER_VALIDATE_EMAIL) === FALSE) {
     //throw exception if email is not valid
     throw new customException($email);
   }
 }
catch (customException $e) {
   //display custom message
   echo $e->errorMessage(); } ?>
```

The "catch" block catches the exception and displays the error message

Web Browser Output



<< THE END >>

PHP Create File – fopen()

PHP File Handling

PHP Create File – fopen()

- The fopen() function is also used to create a file. Maybe a little confusing, but in PHP, a file is created using the same function used to open files.

- If you use fopen() on a file that does not exist, it will create it, given that the file is opened for writing (w) or appending (a).

- The example below creates a new file called "testfile.txt". The file will be created in the same directory where the PHP code resides:

Example

```
$myfile = fopen("testfile.txt", "w")
```

PHP File Permissions

If you are having errors when trying to get this code to run, check that you have granted your PHP file access to write information to the hard drive.

PHP Write to File – fwrite()

PHP File Handling

PHP Write to File – fwrite()

- The fwrite() function is used to write to a file.
- The first parameter of fwrite() contains the name of the file to write to and the second parameter is the string to be written.
- The example below writes a couple of names into a new file called "newfile.txt":

Example

```
<?php
$myfile = fopen("newfile.txt", "w") or die("Unable to open file!");
$txt = "John Doe ";
fwrite($myfile, $txt);
$txt = "Jane Doe ";
fwrite($myfile, $txt);
fclose($myfile);
?>
```

PHP Write to File – fwrite() (Cont.)

Example explained

- Notice that we wrote to the file "newfile.txt" twice. Each time we wrote to the file we sent the string $txt that first contained "John Doe" and second contained "Jane Doe". After we finished writing, we closed the file using the fclose() function.

If we open the "newfile.txt" file it would look like this:

John Doe Jane Doe

PHP Overwriting

- Now that "newfile.txt" contains some data we can show what happens when we open an existing file for writing. All the existing data will be ERASED and we start with an empty file.
- In the example below we open our existing file "newfile.txt", and write some new data into it:

Example

```
<?php
$myfile = fopen("newfile.txt", "w") or die("Unable to open file!");
$txt = "Mickey Mouse ";
fwrite($myfile, $txt);
$txt = "Minnie Mouse ";
fwrite($myfile, $txt);
fclose($myfile);
?>
```

PHP Overwriting (Cont.)

If we now open the "newfile.txt" file, both John and Jane have vanished, and only the data we just wrote is present:

Mickey Mouse Minnie Mouse

PHP File Upload

PHP File Handling

Configure The "php.ini" File

- First, ensure that PHP is configured to allow file uploads.

- In your "php.ini" file, search for the file_uploads directive, and set it to On:

```
file_uploads = On
```

Create The HTML Form

- Next, create an HTML form that allow users to choose the image file they want to upload:

```
<!DOCTYPE html>
<html>
<body>

<form action="upload.php" method="post"
enctype="multipart/form-data">
Select image to upload:
<input type="file" name="fileToUpload" id="fileToUpload">
<input type="submit" value="Upload Image"
name="submit">
</form>

</body>
</html>
```

Create The HTML Form (Cont.)

Some rules to follow for the HTML form are:

- Make sure that the form uses method="post"
- The form also needs the following attribute: enctype="multipart/form-data". It specifies which content-type to use when submitting the form

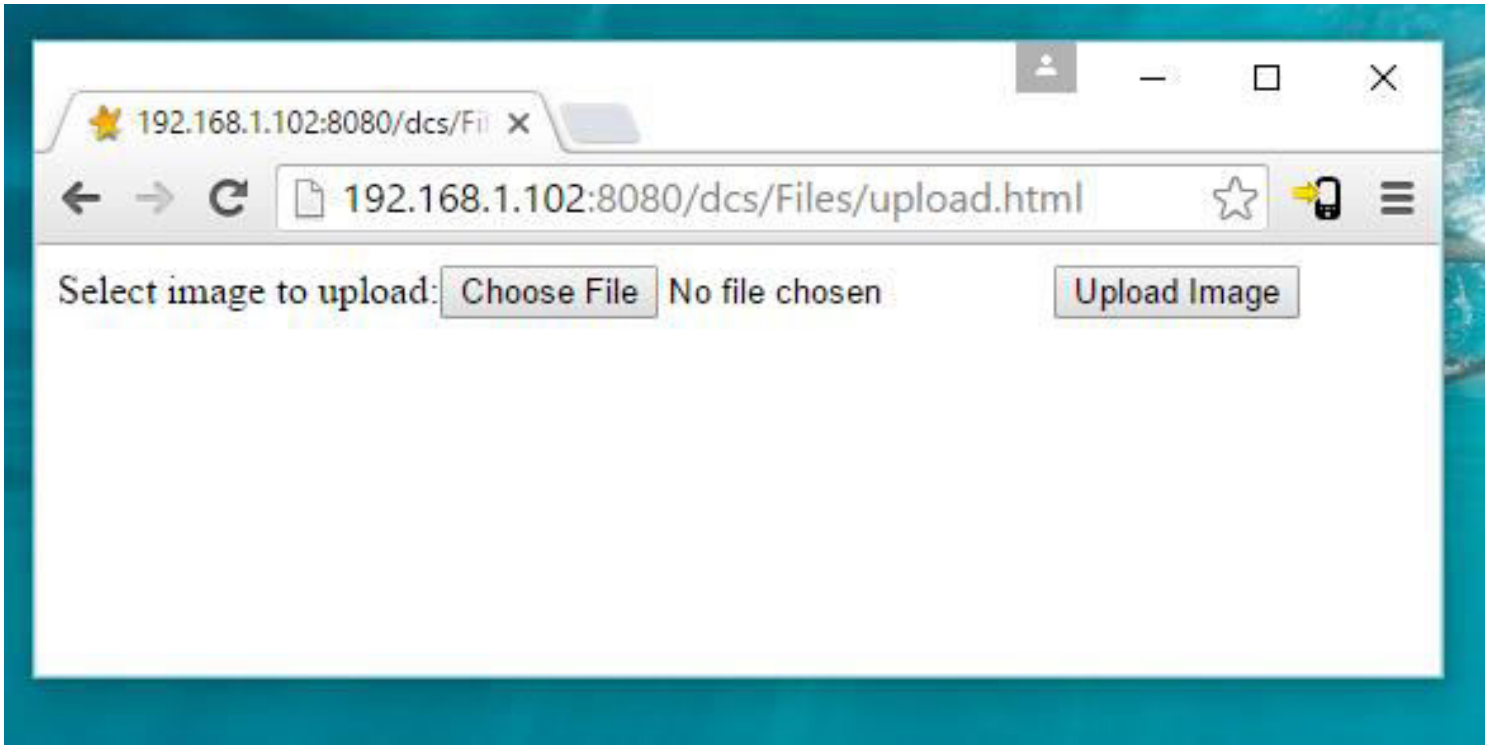
Without the requirements above, the file upload will not work.

Other things to notice:

- The type="file" attribute of the <input> tag shows the input field as a file-select control, with a "Browse" button next to the input control

- The form above sends data to a file called "upload.php", which we will create next.

Web Browser Output



Create The Upload File PHP Script

The "upload.php" file contains the code for uploading a file:

```
<?php
$target_dir = "uploads/";
$target_file = $target_dir . basename($_FILES["fileToUpload"]["name"]);
$uploadOk = 1;
$imageFileType = pathinfo($target_file,PATHINFO_EXTENSION);
// Check if image file is a actual image or fake image
if(isset($_POST["submit"])) {
    $check = getimagesize($_FILES["fileToUpload"]["tmp_name"]);
    if($check !== false) {
        echo "File is an image - " . $check["mime"] . ".";
        $uploadOk = 1;
    } else {
        echo "File is not an image.";
        $uploadOk = 0;
    }
}?>
```

Upload File PHP Script explained

- $target_dir = "uploads/" – specifies the directory where the file is going to be placed
- $target_file specifies the path of the file to be uploaded
- $uploadOk=1 is not used yet (will be used later)
- $imageFileType holds the file extension of the file
- Next, check if the image file is an actual image or a fake image

Note: You will need to create a new directory called "uploads" in the directory where "upload.php" file resides. The uploaded files will be saved there.

Check if File Already Exists

- Now we can add some restrictions.
- First, we will check if the file already exists in the "uploads" folder. If it does, an error message is displayed, and $uploadOk is set to 0:

```
// Check if file already exists
if (file_exists($target_file)) {
    echo "Sorry, file already exists.";
    $uploadOk = 0;
 }
```

Limit File Size

- The file input field in our HTML form above is named "fileToUpload".
- Now, we want to check the size of the file. If the file is larger than 500kb, an error message is displayed, and $uploadOk is set to 0:

```
// Check file size
if ($_FILES["fileToUpload"]["size"] > 500000) {
    echo "Sorry, your file is too large.";
    $uploadOk = 0;
}
```

Limit File Type

- The code below only allows users to upload JPG, JPEG, PNG, and GIF files. All other file types gives an error message before setting $uploadOk to 0:

```
// Allow certain file formats
if($imageFileType != "jpg" && $imageFileType !=
"png" && $imageFileType != "jpeg"
&& $imageFileType != "gif" ) {
    echo "Sorry, only JPG, JPEG, PNG & GIF files are
allowed.";
    $uploadOk = 0;
}
```

Finally Upload File

Now if $uploadOk is set to 0 then file will not be uploaded otherwise file uploading code will be executed

```
// Check if $uploadOk is set to 0 by an error
if ($uploadOk == 0) {
    echo "Sorry, your file was not uploaded.";
// if everything is ok, try to upload file
} else {
    if (move_uploaded_file($_FILES["fileToUpload"]["tmp_name"],
   $target_file)) {
        echo "The file ". basename( $_FILES["fileToUpload"]["name"]). "
   has been uploaded.";
    } else {
        echo "Sorry, there was an error uploading your file.";
    }
}
```

PHP Exception Handling

What is an Exception?

- **Exceptions** are events that occur during the execution of programs that disturb the normal flow of instructions.
- We can also explicitly trigger an exception with throw keyword
- After exception is triggered it must be handled
- This is what normally happens when an exception is triggered:
  - The current code state is saved
  - The code execution will switch to a predefined (custom) exception handler function
  - Depending on the situation, the handler may then resume the execution from the saved code state, terminate the script execution or continue the script from a different location in the code

Resume > skip line

Note: Exceptions should only be used with error conditions, and should not be used to jump to another place in the code at a specified point.

Basic Use of Exceptions

PHP Exception Handling

Basic Use of Exceptions

- When an exception is thrown, the code following it will not be executed, and PHP will try to find the matching "catch" block.
- If an exception is not caught, a fatal error will be issued with an "Uncaught Exception" message.

Lets try to throw an exception without catching it:

```
<?php
 //create function with an exception
 function checkNum($number) {
   if($number>1) {
     throw new Exception("Value must be 1 or below");
   }
   return true;
 }
 //trigger exception
 checkNum(2);
 ?>
```

The code above will get an error like this:

Fatal error: Uncaught exception 'Exception'
with message 'Value must be 1 or below' in C:\webfolder\test.php:6
Stack trace: #0 C:\webfolder\test.php(12):
checkNum(28) #1 {main} thrown in **C:\webfolder\test.php** on line **6**

Try, throw and catch

- To avoid the error from the example above, we need to create the proper code to handle an exception.

- Proper exception code should include:
  1. Try – A function using an exception should be in a "try" block. If the exception does not trigger, the code will continue as normal. However if the exception triggers, an exception is "thrown"
  2. Throw – This is how you trigger an exception. Each "throw" must have at least one "catch"
  3. Catch – A "catch" block retrieves an exception and creates an object containing the exception information

Try, throw and catch example

Lets try to trigger an exception with valid code:

```
<?php
 //create function with an exception
 function checkNum($number) {
   if($number>1) {
     throw new Exception("Value must be 1 or below");   }
   return true;
 }
//trigger exception in a "try" block
 try {
   checkNum(2);
   //If the exception is thrown, this text will not be shown
   echo 'If you see this, the number is 1 or below'; }

//catch exception
 catch(Exception $e) {
   echo 'Message: ' .$e->getMessage();
 }
 ?>
```

The code above will get an error like this:

Message: Value must be 1 or below

Example explained:

- The code throws an exception and catches it:
  1. The checkNum() function is created. It checks if a number is greater than 1. If it is, an exception is thrown
  2. The checkNum() function is called in a "try" block
  3. The exception within the checkNum() function is thrown
  4. The "catch" block retrieves the exception and creates an object ($e) containing the exception information
  5. The error message from the exception is echoed by calling $e->getMessage() from the exception object

WORDPRESS

WHAT IS WORDPRESS

WordPress is web software you can use to create a beautiful website, blog, or app. We like to say that WordPress is both free and priceless at the same time.

Contant management system

WORDPRESS EXAMPLES

- https://wordpress.org/showcase/

BENEFITS OF WORDPRESS

- Free
- Easy to use
- Written in PHP
- MySQL Database at back end
- Easy customization (Themes)
- Simple to understand

THINGS TO KNOW BEFORE INSTALLING WORDPRESS

- Access to your web server
- A text editor
- An FTP Client
- Your web browser of choice

REQUIREMENTS

- You must be able to execute PHP at your web server
- Access to an MySql Database
- In simple you web hosting company should provide PHP and MySQL support

INSTALLING WORDPRESS ON LOCALHOST

- Access to your web server www directory
- A text editor
- Your web browser of choice
- MySQL Database

Table prefix (WordPress++ single DB) wp_

5 Pre Install
lt:9090/wp

DB Name
Username
Password
DB Host ::
Servername

INSTALLING ON REMOTE HOST

Localhost (copy file explore)
Remote (FTP access Upload)

- Access to your web server www or public_html directory
- A text editor
- An FTP Client
- Your web browser of choice
- A MySQL Database

Port number :: MYSQL different location

INSTALLATION PROCESS

Download
Extract
vertrigowww

phpmyadmin
root--vertigo
create DB

5 Pre Install

6 Post Install >

Site Tile
Username
Password
CPassword
Your Email
SEO Visible

ADMINISTRATOR LOGIN

LH:9090/wd

wp-login.php

/wp-admin.p

L name site
R profile pic

Post
Media
Pages
Comment
Appearance
Plugins
User + -
Tools IE
Settings

WORDPRESS DASHBOARD

Theme
Customize
Widget / Menu
Header
Background
Editor

CUSTOMIZATION OF WEBSITE INFORMATION

Stg::General

Site Title
Tagline
URL WP ST
Email
Date -- Time

ADDING PAGES TO YOUR WEBSITE

Page
Title ? DES

A / C / SFP
Static Page

Front Page
Post Page

page attribute -- parent / order

- Find the **Pages** menu in the WordPress Dashboard Navigation menu. Click **Add new.**
- Add the **title** of the page, like *About*. Note: If you have pretty permalinks set up, the title of your page will also be the URL slug.
- Next, add some **content**.
- To Publish the page click on save and publish

USING MEDIA IN WORDPRESS

mp3 >> media player / attachment

- Media consists of the images, video, recordings, and files that you upload and use in your site/blog.

- Media is typically uploaded and inserted into the content when writing a Post or writing a Page.

page >> insert media

max 8mb video -- YT / Compress

MEDIA LIBRARY

The **Media Library Screen** allows you to edit, view, and delete Media previously uploaded to your blog. Multiple Media objects can be selected for deletion.

Search and filtering ability is also provided to allow you to find the desired Media.

Img media -- caption / des / alt / size

manage loc

primary M
social link

Drag more
sub item

HyperLink
> Menu

MENUS IN WORDPRESS

DEFINING A MENU

You must define a menu before you can add items to it.

- Login to the Wordpress Dashboard.
- From the 'Appearance' menu on the left-hand side of the Dashboard, select the 'Menus' option to bring up the Menu Editor.
- Select **Create a new menu** at the top of the page
- Enter a name for your new menu in the Menu Name box
- Click the **Create Menu** button.

ADDING ITEMS TO A MENU

You can add different link types into your menu, these are split between panes left of the menu you're currently editing.

Pans = Section

- Locate the pane entitled **Pages**.
- Within this pane, select the *View All* link to bring up a list of all the currently published Pages on your site.
- Select the Pages that you want to add by clicking the checkbox next to each Page's title.
- Click the **Add to Menu** button located at the bottom of this pane to add your selection(s) to the menu that you created in the previous step.
- Click the **Save Menu** button once you've added all the menu items you want.

DELETING A MENU ITEM

- Locate the menu item that you want to remove in the menu editor window
- Click on the arrow icon in the top right-hand corner of the menu item/box to expand it.
- Click on the *Remove* link. The menu item/box will be immediately removed.
- Click the **Save Menu** button to save your changes.

COMPLETE GUIDE

https://codex.wordpress.org/WordPress_Menu_User_Guide

WORDPRESS THEMES

WordPress themes are simply a group of files, called templates, which determine the look and basic function of your site

FREE VS PAID THEMES

AP / TH / IN / AC

Theme > demand plugin

Install / Activate

FREE THEMES

https://wordpress.org/themes/

PAID THEMES

- http://www.templatemonster.com/wordpress-themes.php
- http://themeforest.net/
- http://www.themezilla.com/themes/

SQL file

WORDPRESS PULGINS

Plugins tiny pieces of software which are used to extend and add to the functionality that already exists in **WordPress**.

The core of **WordPress** is designed to be lean and lightweight, to maximize flexibility and minimize code bloat. **Plugins** then offer custom functions and features so that each user can tailor their site to their specific needs.

php code or plugin

https://wordpress.org/plugins/

Feature > add Shortcode

EXAMPLE

We need to have employee list for our website which we have developed in Wordpress.

We can use plugins like

- Employee Spotlight
- Simple Staff List
- OS Our Team

WORDPRESS THEME CUSTOMIZATION

THEME CUSTOMIZER

The Theme Customizer allows you to preview changes to your site before publishing them. You can also navigate to different pages on your site to preview them.

- Site Title & Tagline
- Colors
- Header Image
- Background Image
- Navigation
- Widgets
- Static Front Page

social URL
hide / unhide
slider -- IDK

FURTHER READING

https://codex.wordpress.org/Appearance_Customize_Screen

<< Finally The End :) >>