Including file & Upload file in PHP

Objectives

- Including file in PHP
- Upload file in PHP

PHP Include Files

- The include (or require) 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.

PHP Include Files

- It is possible to insert the content of one PHP file into another PHP file (before the server executes it), with the include or require statement.
- The include and require statements are identical, except upon failure:
  - require statement will produce a fatal error (E_COMPILE_ERROR) and stop the script
  - include statement will only produce a warning (E_WARNING) and the script will continue

PHP Include Files

- So, if you want the execution to go on and show users the output, even if the include file is missing, use the include statement.
- Otherwise, in case of FrameWork, CMS, or a complex PHP application coding, always use the require statement to include a key file to the flow of execution.
- This will help avoid compromising your application's security and integrity, just in-case one key file is accidentally missing.
- Including files saves a lot of work. This means that you can create a standard header, footer, or menu file for all your web pages. Then, when the header needs to be updated, you can only update the header include file.

PHP Include Files

- Syntax

```
include 'filename';

or

require 'filename';
```

- Example

```
<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 include vs. require

- The require statement is also used to include a file into the PHP code.
- However, there is one big difference between include and require; when a file is included with the include statement and PHP cannot find it, the script will continue to execute,
  the require statement returned a fatal error

```
<html>
<body>

<h1>Welcome to my home page!</h1>
<?php require 'noFileExists.php';
echo "I have a $color $car.";
?>

</body>
</html>
```

Demo

- Teacher demo creating layout of web page by including files

PHP File Uploading

- A PHP script can be used with a HTML form to allow users to upload files to the server. Initially files are uploaded into a temporary directory and then relocated to a target destination by a PHP script.
- An uploaded file could be a text file or image file or any document.
- The process of uploading a file follows these steps
  - The user opens the page containing a HTML form featuring a text files, a browse button and a submit button.
  - The user clicks the browse button and selects a file to upload from the local PC.
  - The full path to the selected file appears in the text filed then the user clicks the submit button.
  - The selected file is sent to the temporary directory on the server.
  - The PHP script that was specified as the form handler in the form's action attribute checks that the file has arrived and then copies the file into an intended directory.
  - The PHP script confirms the success to the user.

Creating an upload script

- There is one global PHP variable called **$_FILES**. This variable is an associate double dimension array and keeps all the information related to uploaded file.
- So if the value assigned to the input's name attribute in uploading form was **file**, then PHP would create following five variables
  - **$_FILES['file']['tmp_name']** − the uploaded file in the temporary directory on the web server.
  - **$_FILES['file']['name']** − the actual name of the uploaded file.
  - **$_FILES['file']['size']** − the size in bytes of the uploaded file.
  - **$_FILES['file']['type']** − the MIME type of the uploaded file.
  - **$_FILES['file']['error']** − the error code associated with this file upload.

Example

```php
<?php
   if(isset($_FILES['image'])){
      $errors= array();
      $file_name = $_FILES['image']['name'];
      $file_size =$_FILES['image']['size'];
      $file_tmp =$_FILES['image']['tmp_name'];
      $file_type=$_FILES['image']['type'];
      $file_ext=strtolower(end(explode('.',$_FILES['image']['name'])));

      $expensions= array("jpeg","jpg","png");

      if(in_array($file_ext,$expensions)=== false){
         $errors[]="extension not allowed, please choose a JPEG or PNG file.";
      }

      if($file_size > 2097152){
         $errors[]='File size must be excately 2 MB';
      }

      if(empty($errors)==true){
         move_uploaded_file($file_tmp,"images/".$file_name);
         echo "Success";
      }else{
         print_r($errors);
      }
   }
?>
<html>
   <body>

      <form action="" method="POST" enctype="multipart/form-data">
         <input type="file" name="image" />
         <input type="submit"/>
      </form>

   </body>
</html>
```

PHP File Uploading

- Some rules to follow for the HTML form above:
  - 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.
  - 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

Demo

- Teacher demo code about uploading file process for students.