Cómo subir archivos al servidor con PHP

Hace poco hablábamos de [enviar formularios al servidor usando ajax](http://www.enricflorit.com/como-enviar-formularios-con-ajax-y-formdata/). Este método es muy útil siempre y cuando no tengas que subir archivos, ya que no los soporta. Así que vamos a tocar el tema de subir archivos al servidor usando formularios. Hoy le toca a PHP, y más adelante hablaré del tema usando AJAX.

Para el proceso usaremos dos archivos. Se podría hacer con uno solo, pero sería complicarnos la vida comprobando si se han enviado archivos previamente, etc. El primero de los archivos puede ser tanto **\*.html** como **\*.php**. El segundo tiene que ser obligatoriamente **.php**, para que el servidor lo interprete.

El código html

El primer archivo (vamos a llamarlo index.html, o como queráis) simplemente contendrá un formulario. El método del formulario será POST. También necesitaremos un atributo poco común, **enctype**. Esto hará que al enviar el formulario, la página que lo recibirá ya sabrá que le estamos enviando datos de un formulario (multipart/form-data).

Dentro del formulario, un **input** de tipo **file** (archivo) y un botón. Ya lo tenemos definido, vamos con el código:



|  |  |
| --- | --- |
| 1  2  3  4  5  6 | <form action="subir\_archivo.php" method="post" enctype="multipart/form-data">    <label for="archivo">Archivo:</label>    <input type="file" name="archivo" id="archivo" />    <br/>    <input type="submit" value="Enviar" />  </form> |

Con esto tendremos listo nuestro código HTML. Vamos a pasar al PHP…

Comprobando si se ha cargado un archivo

Para evitar errores “tontos”, vamos a comprobar antes que nada si realmente se ha enviado un archivo. Si no es así, se avisará al usuario de que debe ir a la página HTML para cargar un archivo.



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | <?php    if( !isset($\_FILES['archivo']) ){    echo 'Ha habido un error, tienes que elegir un archivo<br/>';    echo '<a href="index.html">Subir archivo</a>';  }else{    //Nuestro código aquí  } |

Manejando posibles errores y mostrando la información del archivo cargado

Lo siguiente será manejar los posibles errores: puede que no se haya subido el archivo, por ejemplo. Así que dentro del bloque **else** escribiremos el siguiente código:



|  |  |
| --- | --- |
| 1  2  3 | if( $\_FILES['archivo']['error'] > 0 ){    echo 'Error: ' . $\_FILES['archivo']['error'] . '<br/>';  } |

También mostraremos la información del archivo subido. Puede que en nuestra aplicación no vayamos a hacer visible estos datos, pero saber cuáles son los disponibles será útil para luego poder guardar el archivo. Los datos son: nombre, tipo de archivo, tamaño (se proporciona en bytes, lo pasaremos a Kb) y la ruta temporal.



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | if( $\_FILES['archivo']['error'] > 0 ){    echo 'Error: ' . $\_FILES['archivo']['error'] . '<br/>';  }else{    echo 'Nombre: ' . $\_FILES['archivo']['name'] . '<br/>';    echo 'Tipo: ' . $\_FILES['archivo']['type'] . '<br/>';    echo 'Tamaño: ' . ($\_FILES['archivo']['size'] / 1024) . ' Kb<br/>';    echo 'Guardado en: ' . $\_FILES['archivo']['tmp\_name'];  } |

Guardando el archivo

Ya solo nos queda comprobar si el archivo no es demasiado grande, así como su tipo (a veces coincide con la extensión; p. ej.: .html, .gif, .jpg…). Por ejemplo, si queremos que solo se puedan subir imágenes, deberemos permitir las siguientes extensiones: **jpg**, **jpeg**, **gif** y **png**.

Para saber la extensión y comprobar si está permitida tendremos que hacer un **explode** del nombre del archivo y comparar la última parte con el array de extensiones usando **end**.



|  |  |
| --- | --- |
| 1  2  3 | $ext\_permitidas = array('jpg','jpeg','gif','png');  $extension = end( explode('.', $\_FILES['archivo']['name']) );  $ext\_correcta = in\_array($extension, $ext\_permitidas); |

También tenemos que comprobar el tipo del archivo. Usaremos expresiones regulares:



|  |  |
| --- | --- |
| 1 | $tipo\_correcto = preg\_match('/^image\/(pjpeg|jpeg|gif|png)$/', $\_FILES['archivo']['name']); |

Si queremos, podemos limitar el tamaño del archivo, algo muy recomendable. Nadie tiene un servidor con una capacidad ilimitada, ni tampoco un internet tan rápido como para subir imágenes de cientos de megas. Por ejemplo, para poner un límite de 500 Kb:



|  |  |
| --- | --- |
| 1 | $limite = 500 \* 1024; |

Ahora solo nos queda juntar todas estas condiciones y guardar el archivo, si las cumple. Lo guardaremos en una carpeta a la que llamaremos **subidas**, que estará en el mismo directorio que el **index.html** y el **subir\_archivo.php**.



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21 | if( $ext\_correcta && $tipo\_correcto && $\_FILES['archivo']['size'] <= $limite ){    if( $\_FILES['archivo']['error'] > 0 ){      echo 'Error: ' . $\_FILES['archivo']['error'] . '<br/>';    }else{      echo 'Nombre: ' . $\_FILES['archivo']['name'] . '<br/>';      echo 'Tipo: ' . $\_FILES['archivo']['type'] . '<br/>';      echo 'Tamaño: ' . ($\_FILES['archivo']['size'] / 1024) . ' Kb<br/>';      echo 'Guardado en: ' . $\_FILES['archivo']['tmp\_name'];        if( file\_exists( 'subidas/'.$\_FILES['archivo']['name']) ){        echo 'El archivo ya existe: ' . $\_FILES['archivo']['name'];      }else{        move\_uploaded\_file($\_FILES['archivo']['tmp\_name'],          "subidas/" . $\_FILES['archivo']['name']);          echo "Guardado en: " . "subidas/" . $\_FILES['archivo']['name'];      }    }  }else{    echo 'Archivo inválido';  } |

Código final

Así es como quedaría nuestro código PHP al completo, probado (con algunas modificaciones de última hora):



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43 | <?php    if( !isset($\_FILES['archivo']) ){    echo 'Ha habido un error, tienes que elegir un archivo<br/>';    echo '<a href="index.html">Subir archivo</a>';  }else{      $nombre = $\_FILES['archivo']['name'];    $nombre\_tmp = $\_FILES['archivo']['tmp\_name'];    $tipo = $\_FILES['archivo']['type'];    $tamano = $\_FILES['archivo']['size'];      $ext\_permitidas = array('jpg','jpeg','gif','png');    $partes\_nombre = explode('.', $nombre);    $extension = end( $partes\_nombre );    $ext\_correcta = in\_array($extension, $ext\_permitidas);      $tipo\_correcto = preg\_match('/^image\/(pjpeg|jpeg|gif|png)$/', $tipo);      $limite = 500 \* 1024;      if( $ext\_correcta && $tipo\_correcto && $tamano <= $limite ){      if( $\_FILES['archivo']['error'] > 0 ){        echo 'Error: ' . $\_FILES['archivo']['error'] . '<br/>';      }else{        echo 'Nombre: ' . $nombre . '<br/>';        echo 'Tipo: ' . $tipo . '<br/>';        echo 'Tamaño: ' . ($tamano / 1024) . ' Kb<br/>';        echo 'Guardado en: ' . $nombre\_tmp;          if( file\_exists( 'subidas/'.$nombre) ){          echo '<br/>El archivo ya existe: ' . $nombre;        }else{          move\_uploaded\_file($nombre\_tmp,            "subidas/" . $nombre);            echo "<br/>Guardado en: " . "subidas/" . $nombre;        }      }    }else{      echo 'Archivo inválido';    }  } |

Espero que os pueda servir este script de carga de archivos. A partir de aquí, se podría desarrollar mucho, por ejemplo, usar bases de datos para clasificar los archivos, asignarselos a usuarios…

**Si te gustó este post y te resultó útil, no olvides escribirlo abajo en los comentarios y seguirme en twitter**[**@enricflorit**](http://twitter.com/enricflorit)**. Hasta pronto! ;)**

- See more at: http://www.enricflorit.com/como-subir-archivos-al-servidor-con-php/#sthash.jS6BK7NE.dpuf