1 Curso POO PHP Formularios web

1.1 Formularios web

A forma natural para facer chegar á aplicación web os datos do usuario dende un navegador, é utilizar formularios HTML. Os formularios HTML van encerrados sempre entre as etiquetas **<FORM> <**/**FORM>**. Dentro dun formulario inclúense os elementos sobre os que pode actuar o usuario, principalmente usando as etiquetas **<INPUT>**, **<SELECT>**, **<TEXTAREA>** e **<BUTTON>**.

O atributo **action** do elemento FORM indica a páxina á que se lle enviarán os datos do formulario. No noso caso tratarase dun script PHP. Pola súa banda, o atributo **method** especifica o método usado para enviar a información. Este atributo pode ter dous valores:

- **get**: con este método os datos do formulario agréganse ao URI utilizando un signo de consulta **"?"** como separador.
- **post**: con este método os datos inclúense no corpo dunha mensaxe de petición POST de HTTP.

En PHP os datos recolleranse de distinta forma dependendo de como se envíen. O datos enviados co método POST recóllense empregando a variable **$_POST**.

```
<?php
    $nome = $_POST['nome'];
    $modulos = $_POST['modulos'];
    print "Nome: ".$nome."<br />";
    foreach ($modulos as $modulo) {
        print "Modulo: ".$modulo."<br />";
    }
?>
```

Se pola contra tiveses usado o método GET, o código necesario para procesar os datos sería similar; simplemente faría falta cambiar a variable $_POST por **$_GET**.

```
<?php
    $nome = $_GET['nome'];
    $modulos = $_GET['modulos'];
    print "Nome: ".$nome."<br />";
    foreach ($modulos as $modulo) {
        print "Modulo: ".$modulo."<br />";
    }
?>
```

En calquera dos dous casos poderías ter usado **$_REQUEST** substituíndo respectivamente $_POST e $_GET. $_REQUEST almacena o contido dos arrays $_POST, $_GET e $_COOKIE.

```
<?php
    $nome = $_REQUEST['nome'];
    $modulos = $_REQUEST['modulos'];
    print "Nome: ".$nome."<br />";
    foreach ($modulos as $modulo) {
        print "Modulo: ".$modulo."<br />";
    }
?>
```

1.1.1 Empregar un formulario web para subir ficheiros ao servidor

É posible empregar o método POST para subir un ficheiro ao servidor web. Neste caso é necesario que o tipo de codificación para os datos que se envían estea especificado como "*multipart/form-data*" na etiqueta "*<form>*", e crear unha etiqueta HTML de tipo "*<input>*" co atributo "*type='file'*".

```
<form enctype="multipart/form-data" action="envio.php" method="POST">
    Ficheiro a enviar: <input name="ficheiro" type="file" />
    <input type="submit" value="Enviar" />
</form>
```

O nome do arquivo enviado recíbese no array global **$_FILES**, e unha vez recibido pode moverse a unha nova localización empregando a función **move_uploaded_file**. A localización temporal na que se reciben os ficheiros subidos polos clientes web e outros datos relativos ás subidas poden xestionarse empregando as seguintes directivas de configuración:

- **post_max_size**. Tamaño máximo da información transmitida co método POST. Afecta aos ficheiros e a calquera outra información transmitida por ese método.

- **file_uploads**. Indica se se permiten ou non as subidas de ficheiros ao servidor.

- **upload_max_filesize**. Tamaño máximo de cada un dos ficheiros que se suben ao servidor.

- **upload_tmp_dir**. Directorio no que se almacenan de forma temporal os ficheiros que se suben ao servidor web.

Aos límites de tamaño de ficheiro anteriores aplícanselles a maiores aqueles que teña configurados o servidor web, se é o caso.

Se o ficheiro se sube correctamente ao servidor, crearase un novo elemento no array *$_FILES* co nome definido no atributo "*name*" do elemento "*<input>*", e cos seguintes membros.

- **$_FILES['ficheiro']['name']**. Nome orixinal do ficheiro.
- **$_FILES['ficheiro']['size']**. Tamaño do ficheiro.
- **$_FILES['ficheiro']['tmp_name']**. Nome temporal do ficheiro recibido no servidor.
- **$_FILES['ficheiro']['error']**. Código de erro no caso dun envío incorrecto.

--Víctor Lourido 12:07 15 jul 2013 (CEST)