Formularios y recogida de datos

Índice de contenidos

Podemos enviar un formulario a través de los method POST y GET. El action determinará la página de destino, que puede ser la misma página, otra o bien una llamada a funciones, como por ejemplo de javascript

GET

Ejemplo:

```
<form action="/action_page.php" method="get">
  <label for="fname">First name:</label>
  <input type="text" id="fname" name="fname"><br>
  <label for="lname">Last name:</label>
  <input type="text" id="lname" name="lname"><br>
  <input type="submit" value="Submit">
</form>
```

Si rellenamos el formulario con los valores Raquel y Sanz y lo enviamos, en el navegador nos aparecerá lo siguiente:

URL_DOMINIO/action_page.php?fname=Raquel&lname=Sanz

Para recoger esos datos en action_page.php deberemos utilizar la variable superglobal $_GET

```
<?php
$nombre  = $_GET[“fname”];
$apellido  = $_GET[“lname”];

echo “Tu nombre completo es $nombre $apellido”;
```

Podemos utilizar get cuando no sea necesario ocultar los valores que se pasan a la página de destino o queramos poder compartir un enlace con unos parámetros concretos.

POST

Ejemplo:

```
<form action="/action_page.php" method="post">
```

```
  <label for="fname">First name:</label>
  <input type="text" id="fname" name="fname"><br><br>
  <label for="lname">Last name:</label>
  <input type="text" id="lname" name="lname"><br><br>
  <input type="submit" value="Submit">
</form>
```

Si rellenamos el formulario con los valores Raquel y Sanz y lo enviamos, en el navegador nos aparecerá lo siguiente:

URL_DOMINIO/action_page.php

En este caso los parámetros pasados no son visibles.

Para recoger esos datos en action_page.php deberemos utilizar la variable superglobal $_POST

```
<?php
$nombre  = $_POST[“fname”];
$apellido  = $_POST[“lname”];

echo “Tu nombre completo es $nombre $apellido”;
```

Podemos utilizar get cuando sea necesario ocultar los valores que se pasan a la página de destino y/o queramos acceder a una página destino sin unos parámetros preestablecidos.

Validación en destino

Necesitamos validar en el PHP de destino la información que se pasa a través de get o post, ya que:

- Podemos recibir información incorrecta que puede dar errores
- Podemos estar siendo atacados con código malicioso

Parámetro vacío o inexistente

Para validar que un parámetro se está recibiendo correctamente podemos comprobar si existe y/o si está vacía.

Si solamente queremos comprobar si existe aunque esté vacía podemos utilizar la función **isset** de PHP. Si queremos comprobar si no existe y/o está vacía, podremos utilizar la función **empty**.

A veces, un parámetro puede incluir espacios en blanco, tabuladores u otros caracteres que nosotros consideraríamos vacíos. Para eliminarlos, podemos utilizar la función **trim**, que elimina los espacios en blanco y saltos de línea entre otros del inicio y final del string. También puede eliminar del principio y del final otros strings, pasando estos como segundo parámetro.

Ejemplo de trim:

```php
<?php
$cadena = "  texto de ejemplo   ";
$cadena_formateada = trim($cadena);

echo "La cadena original es: '".$cadena."' y la formateada es:
'".$cadena_formateada."'";

// Salida: La cadena original es: '  texto de ejemplo    ' y la
formateada es 'texto de ejemplo'
```

Evitando inyecciones de código

Para evitar inyecciones de código por parte de atacantes, podemos filtrar las / que nos envíen a la página de destino gracias al método stripslashes y convertir los caracteres especiales a entidades html a través de htmlspecialchars.

Ejemplo de stripslashes:

```php
<?php
$str = "Is your name O\'reilly?";
// Salida: Is your name O'reilly?
echo stripslashes($str);
```

```
?>
```

Ejemplo de htmlspecialchars:

```
<?php
$new = htmlspecialchars("<a href='test'>Test</a>", ENT_QUOTES);
echo $new;
// Lo convierte en:
// &lt;a href=&#039;test&#039;&gt;Test&lt;/a&gt;
?>
```

Valores en los rangos esperados

Una vez hemos validado que la variable existe y la hemos “limpiado” de posibles ataques y caracteres no deseados, debemos validar las características que tiene que tener nuestra variable.

Por ejemplo, si recibimos una variable de año de nacimiento debemos comprobar que sea una fecha correcta y que sea anterior al día de hoy.