Programación Avanzada
Grado en Ingeniería Informática en Sistemas de Información - Curso 2018/2019

EPD 4: Procesamiento de formularios en PHP y funciones de utilidad

La entrega del trabajo se hará a través de la tarea correspondiente en el Campus Virtual. Pasado el límite de entrega se aceptará el envío del trabajo, con una penalización de 2 puntos sobre 10 de la calificación por cada hora o fracción de retraso. La entrega consistirá en un único fichero comprimido en formato ZIP cuyo nombre deberá ser de la forma *equipoXX.zip*, donde *XX* serán dos cifras que indicará el número del equipo. Por ejemplo, equipo07.zip. Este fichero contendrá una serie de carpetas cuyo nombre deberá ser de la forma *ejY* o *pZ*, donde *Y* y *Z* representan, respectivamente, el número de cada ejercicio o problema del presente guión. Dentro de dichas carpetas se incluirán exclusivamente los archivos necesarios para la resolución del correspondiente ejercicio o problema. Las rutas de los ficheros empleados serán relativas, a fin de que las resoluciones a los ejercicios y problemas puedan ser examinadas en cualquier equipo. Cualquier entrega que no cumpla las reglas de nombrado, el formato de compresión del archivo o el contenido de los archivos del mismo, será penalizada con 2 puntos sobre 10 por cada incumplimiento.

Objetivos

- Crear código PHP que procese datos procedentes de formularios.
- Conocer y emplear las funciones de utilidad incluidas en PHP.

Conceptos

1. Procesamiento de formularios en PHP

PHP ofrece un mecanismo muy sencillo y rápido para el acceso, y posterior procesamiento, de los valores introducidos por el usuario mediante un formulario HTML. Este mecanismo se basa en el uso de las variables superglobales *$_GET* y *$_POST* (según sea, respectivamente, *get* o *post* el método empleado por el navegador para el envío de los datos de los campos, indicado en el atributo *method* del formulario) así como *$_REQUEST* (que es independiente del método de envío empleado). No obstante, esta última, puede incorporar valores de las *cookies* transmitidas por el navegador y depende, en última instancia, de la configuración de PHP específica para el servidor, por lo que no es muy recomendable utilizarlo.

Estas variables superglobales son vectores asociativos que contienen un elemento por cada campo del formulario cuya acción (atributo *action*) coincida con la página PHP en cuestión. Estos elementos tendrán asociado como clave la cadena de caracteres correspondiente al nombre dado al campo en el formulario (atributo *name*). El valor asociado a cada elemento del vector será el introducido por el usuario mediante teclado (en caso de campos textuales) o el asignado a la opción seleccionada (listas desplegables, botones tipo radio, casillas de verificación / *checkboxes* y botones de envío) para el campo del formulario cuyo nombre coincide con la clave de dicho elemento.

Puede consultar más detalles sobre el procesamiento de formularios de PHP en el capítulo 6 de [1].

2. Funciones de utilidad

Hasta el momento, en las sesiones de enseñanzas básicas ha estudiado funciones de utilidad para el manejo de vectores y cadenas de caracteres. Encontrará material de referencia rápida sobre estas y otras funciones de utilidad incluidas en PHP en el Apéndice B de [1]. Para un análisis más detallado de cada una de las funciones se recomienda la consulta de [2] y, concretamente según el tipo de función, de [3] y [4].

Bibliografía Básica

1. PHP 5: fast & easy web development. Julie Meloni. Thomson Course Technology, 2004. Parte II. http://0-site.ebrary.com.athenea.upo.es/lib/bupo/Doc?id=10058862
2. Manual de PHP. http://www.php.net/manual/en/index.php
3. Referencia de funciones de utilidad para vectores y matrices. http://www.php.net/manual/en/ref.array.php
4. Referencia de funciones de utilidad para cadenas de caracteres. http://www.php.net/manual/en/ref.strings.php

Experimentos

E1. (30 mins.) Cree un formulario con varios campos y analice el tráfico entre el navegador y su servidor. Puede crear para ello una página PHP que no realice ningún procesamiento y realizar el análisis de tráfico empleando el inspector de Chrome.

a) ¿Qué tráfico se intercambian navegador y servidor? ¿Cuál es su significado?

b) Haga varios envíos alternando el método ¿Observa alguna diferencia en el tráfico enviado?

c) Incluya un campo de contraseña ¿Se puede considerar seguro? ¿De qué forma puede detectar el contenido?

E2. (30 mins.) Cree, dentro de un nuevo proyecto PHP de NetBeans, un archivo de PHP y emplee el siguiente código:

```
<?php
function test_alterar ($elemento, $clave, $prefijo) {
    echo "Elemento original: $elemento <br /> \n";
    $elemento = "$prefijo: $elemento";
    echo "Elemento modificado: $elemento <br /> \n";
}

function test_ver ($elemento, $clave) {
    echo "$clave. $elemento<br />\n";
}
?>
<html>
    <body>
        <?php
        $frutas = array ("d"=>"limon", "a"=>"naranja", "b"=>"platano", "c"=>"manzana");

        echo "Antes...: <br />\n";
        array_walk ($frutas, 'test_ver');

        echo "<br />Alteramos: <br /> \n";
        array_walk ($frutas, 'test_alterar', 'fruta');

        echo "<br />...y despues: <br /> \n";
        array_walk ($frutas, 'test_ver');
        ?>
    </body>
</html>
```

a) ¿Cuál es el objetivo del código? ¿Cuál es el cometido de la función *array_walk*? Busque su documentación en [2] y analícela para entender dicho cometido y la forma de uso de la función.

b) Ejecute la página PHP propuesta ¿Es la salida la esperada? ¿Por qué? Solucione el problema.

Ejercicios

EJ1. (50 mins.) Cree una página PHP que recoja a través de un formulario información sobre las carreras de Moto GP. El formulario permitirá introducir el nombre de la carrera (Jerez, Motegi, Sepang, Montmeló,…) y la fecha de la misma (formato: dd/mm/aaaa). También permitirá elegir entre "Vuelta rápida" o "Vencedor" mediante un botón tipo "radio". Si se elige "Vuelta rápida" la página mostrará en una tabla (con encabezados descriptivos) el nombre del piloto, el n.º de vuelta y el tiempo de vuelta mínimo de todo el GP (es decir, aquella vuelta en el que el tiempo de vuelta es el menor de todos los del GP). Si se elige "Vencedor" la página mostrará en una tabla (con encabezados descriptivos) el nombre del piloto ganador del GP (es decir, el que completó el conjunto de vueltas en un tiempo menor) y el tiempo total del mismo para todo el GP. Cada GP tendrá un mínimo de 3 y un máximo de 50 vueltas. En un área de texto recogerá una lista de los tiempos por vuelta de la carrera de Moto GP. En cada línea se leerá la siguiente información: N.º de vuelta (natural), nombre del piloto (cadena) y tiempo por vuelta (minutos:segundos:centénsimas). A continuación se muestra un ejemplo:

```
1;              Pedrosa;        3:15:55
1;              Marc Marquez;   3:15:60
1;              Dovicioso;      3:15:30
1;              Rossi;          3:16:03
1;              Lorenzo;        3:14:95
1;              Zarco;          ::
…
27;             Pedrosa;        3:14:90
27;             Marc Marquez;   3:14:60
27;             Dovicioso;      ::
27;             Rossi;          3:15:30
27;             Lorenzo;        3:15:09
…
```

Se observa que un abandono se indica con :: en el tiempo por vuelta, el separador de campos es el ';'.

En la ampliación de bibliografía [5] y [6] se han añadido enlaces con información sobre los pilotos y circuitos de moto GP.

EJ2. (15 mins.) Modifique la página del ejercicio anterior para que el área de texto empleada permita introducir también el equipo de los pilotos y la página permita mostrar el equipo ganador (aquel cuyo suma de tiempos de sus pilotos es menor). Cada equipo tiene 2 pilotos. Los circuitos y fechas de carrera se elegirán mediante un campo tipo *select*. Además se podrá mostrar tanto la vuelta rápida como el vencedor (campo tipo *checkbox*). En la ampliación de bibliografía [7] se han añadido enlaces con información sobre los equipos de moto GP. La lista de equipos deberá mostrarse ordenada de menor a mayor tiempo de equipo.

Problemas

P1. (35 mins.) Cree una página PHP que permita señalar frases demasiado cortas o largas en un texto. Para ello, se ofrecerá al usuario un cuadro de texto donde podrá incluir el texto a revisar, además de dos deslizadores que permiten seleccionar dos valores numéricos, ambos deben estar entre 1 y 10. Estos dos valores serán los umbrales que expresan el número de palabras mínimo y máximo para considerar una frase demasiado corta o demasiado larga en el texto, respectivamente. En caso de que alguna de las frases que contiene el texto a revisar no esté dentro de los límites, se imprimirá el texto, marcando las frases con un formato de énfasis (uno distinto para frases cortas y largas) y se indicará que el texto contiene errores. En caso contrario se imprimirá el texto y el mensaje indicará que es correcto.

P2. (40 mins.) Mejore el sistema de detección de frases cortas/largas desarrollado en el ejercicio anterior de tal forma que solvente automáticamente las frases que exceden los límites. Para ello, dispondremos de un vector de palabras que servirá para rellenar texto en las frases cortas. En concreto, las palabras del vector se irán intercalando entre palabra y palabra en las frases cortas hasta que éstas alcancen el umbral mínimo. Las frases largas, por su parte, deberán ser acortadas eliminando todas las palabras que sobren e introduciendo puntos suspensivos (...) al final de la frase. El sistema deberá mostrar el texto original y el texto al que se le han aplicado las correcciones automáticas, marcando éstas con un formato de énfasis (distinto al empleado en el ejercicio anterior).

P3. (50 mins.) Añada a la página PHP de los ejercicios 1 y 2 un nuevo formulario que se muestre al cargarla por primera vez. En este formulario se recogerán los circuitos o carreras de MotoGP de todo el campeonato (un mínimo de 3) y permitirá conocer el campeón del campeonato de Moto GP, mostrándolo en una tabla. Posteriormente deberán introducirse los datos incluidos en los ejercicios 1 y 2 para el número de carreras indicado. Se aconseja introducir un campo oculto que permita conocer el n.º de formularios de datos de carrera que habrá que mostrar y procesar. Una vez fijado el n.º de carreras, la información se introducirá en campos individuales en lugar de mediante un área de texto. Se recomienda aplicar programación modular, dividiendo la funcionalidad de la página en varias funciones distintas.

P4. (90 mins.) Mejore la página desarrollada en el problema 5 del guión de la EPD 3, de tal forma que se cree un interfaz que permita la introducción de los datos de entrada (matriz y escalar), empleando una tabla en el caso de la matriz. Para ello, al invocar por primera vez la página se pedirá al usuario que indique las dimensiones de la matriz. Una vez enviado este dato se le ofrecerá al usuario un formulario con un cuadro de texto para el escalar y la tabla que permita introducir los datos de la matriz a procesar. Al recibir estos datos, se deberá comprobar que todos los datos se encuentran rellenos y que son numéricos (emplee para ello la función is_numeric[1]). En caso de error, se volverá a mostrar el formulario de nuevo con los datos correctos ya prerrellenos y los datos incorrectos en sus campos teniendo éstos un fondo rojo para indicar el error. Este proceso se repetirá indefinidamente hasta que no haya errores. Se recomienda dividir la funcionalidad de la página en funciones a fin de obtener un código más legible.

1 Puede obtener más información sobre esta función en http://es.php.net/is_numeric.

Ampliación de Bibliografía

1. PHP 5 Power Programming. Andi Gutmans, Stig Bakken, Derick Rethans. Prentice Hall, 2004. Capítulo 2
   http://www.informit.com/content/images/013147149X/downloads/013147149X_book.pdf
2. Beginning PHP5, Apache, and MySQL Web Development. Elizabeth Naramore, Jason Gerner, Yann Le Scouarnec. Willey, 2005. Capítulos 1 y 2.
   http://site.ebrary.com/lib/bupo/docDetail.action?docID=10114243
3. Beginning PHP5, Apache, and MySQL Web Development. Elizabeth Naramore, Jason Gerner, Yann Le Scouarnec. Willey, 2005. Capítulo 5 y apéndice D.
   http://site.ebrary.com/lib/bupo/docDetail.action?docID=10114243
4. PHP 5 for dummies. Janet Valade. Wiley Pub., 2004. Capítulos del 3 al 8.
   http://0-site.ebrary.com.athenea.upo.es/lib/bupo/Doc?id=10114230
5. Información sobre MotoGP: Circuitos
   http://www.motogp.com/es/calendar
6. Información sobre MotoGP: Pilotos
   http://www.motogp.com/es/riders/MotoGP
7. Información sobre MotoGP: Equipos
   http://www.motogp.com/es/teams/MotoGP

Datos de la Práctica

Autor del documento: Carlos D. Barranco González (Diciembre 2007).

Revisiones:

1. Carlos D. Barranco González (Enero 2008).
2. Carlos D. Barranco González (Noviembre 2009). Adaptación a nueva planificación, revisión de texto y de enlaces.
3. Carlos D. Barranco González (Noviembre 2010). Revisión del texto, adaptación de formato de cabecera y renovación de los ejercicios 3 y 4 , así como una ligera modificación del ejercicio 2 y el problema 1.
4. Alejandro Gómez Morón (Noviembre 2011): Renovación ejercicios 1 y 2, modificación media del problema 1, así como ligera modificación del problema 2.
5. Carlos D. Barranco González (Noviembre 2011): Revisión del texto.
6. Carlos D. Barranco (Noviembre 2012): Adaptación del guión a la asignatura Programación Avanzada. Actualización de la referencia bibliográfica 3 del apartado de ampliación de la bibliografía. Corrección del texto en la sección "Conceptos" y corrección del formato en los ejercicios.
7. Miguel A. Montero (Octubre 2013): Actualización ampliación de bibliografía, ítems 2 y 3. Actualización del problema 2.
8. Carlos D. Barranco (Octubre 2013): Mejoras y correcciones en la sección de Conceptos. Mejora de las cuestiones del experimento 1. Mejora de la redacción de Ej2, Ej4, P1 y P2. Corrección de formato del item 1 de la bibliografía de ampliación.
9. Miguel A. Montero (Noviembre 2013): Renumeración ejercicios 1 a 4 por 2 a 5. Redacción ejercicio 1, modificación temporización ejercicio 3 y 5. Actualización del problema 1.
10. Carlos D. Barranco (Noviembre 2013): Mejora de la redacción del ejercicio 1. Cambio en la redacción del problema 1 para evitar referencias a datos guardados en disco.
11. Miguel A. Montero (Noviembre 2013): Eliminación ejercicio 1. Renumeración ejercicios 2 a 5 por 1 a 4. Corrección referencia a pie de página del problema 2.
12. Carlos D. Barranco (Octubre 2014): Mejora del texto de la sección de conceptos. Introducción del experimento 1. Nueva estimación de tiempo para el experimento 2 y el resto del guión. Renovación de los enunciados de EJ1, EJ2, P3 y P4.
13. Carlos D. Barranco (Octubre 2015): Mejoras en el texto de la sección de conceptos. Corrección del texto en Experimento 1. Cambios de enunciado en los ejercicios 1 y 2, así como en el problema 3. Arreglada referencia que hecha en el problema 5 a un ejercicio del guión anterior.
14. Jose A. Gómez (Octubre 2016): Cambio del enunciado de los ejercicios 1 y 2 y del problema 3.
15. Carlos D. Barranco (Octubre 2016): Adiciones en la parte de conceptos.
16. Gualberto Asencio (Octubre 2017): Corrección errata leve en sección de conceptos. Modificados enunciados de los problemas 1, 2 y 4.
17. Carlos D. Barranco (Octubre 2017): Corrección de errata en enunciado del ejercicio 1.
18. Pedro Espina (Octubre 2018): actualización de año y pequeños cambios de redacción. Se han actualizado los enunciados de los ejercicios 1 y 2, así como del problema 3. También se ha cambiado el reparto de tiempos entre el ejercicio 1 y el problema 3.

Estimación temporal:

- Parte presencial: 120 minutos.
  - Explicación inicial: 5 minutos.
  - Experimentos: 60 minutos.
  - Ejercicios: 55 minutos.
- Parte no presencial: 270 minutos.
  - Lectura y estudio del guión y bibliografía básica: 45 minutos
  - Problemas: 225 minutos