



DOCUMENTACIÓN PROYECTO 1

Etapas 5-7

Ismael Aliaga Molina

080 Formación

1º Desarrollo de Aplicaciones Web

Tabla de contenido

Petición del cliente	3
Etapas 5.....	4
Etapas 6.....	7
Etapas 7.....	9
Herramientas utilizadas	12
Conclusión	14

Documentación Proyecto 1

Petición del cliente

INTRODUCCIÓN

Nuestro cliente se ha quedado conforme con la primera versión que se le ha entregado, cumplía con sus necesidades iniciales.

Al utilizar la aplicación se ha dado cuenta de que le vendría bien que le hiciésemos las siguientes nuevas funcionalidades:

- Crear las tareas pendientes desde la aplicación web
- Mostrar las tareas ordenados por id

REQUISITOS DEL PROYECTO

- Crear las tareas **pendientes** desde la aplicación web
- Las tareas se muestren ordenadas por id
 - Cuando cree una nueva tarea pendiente, podrá especificar el id. Para ello:
 - El id que especifique no debe existir ya entre las tareas existentes
 - Si no especifica el id, se creará con el siguiente id que corresponda
- Para garantizar que los id's son correctos, se hará un validador que compruebe que los id's comienzan en 0, no se repite ninguno y no existen huecos.
- Si se quiere modificar el id de las tareas pendientes, el usuario lo realizará de forma manual. *(Se omite este requisito en la documentación ya que no hay que realizar nada)*
- El id de las tareas en progreso y terminadas, se asignará de forma manual. *(Se omite este requisito en la documentación ya que no hay que realizar nada)*

Etapa 5

INTRODUCCIÓN

En esta etapa vamos a abarcar el siguiente requisito del proyecto:

- Las tareas se muestren ordenadas por id

COMO ORGANIZAR LAS TAREAS POR ID

Ahora mismo tenemos el fichero pendientes.txt con el siguiente contenido:

Pendientes	En proceso	Finalizadas
Tarea 2	Tarea 2	Tarea 2
Tarea 3	Tarea 3	Tarea 3
Tarea 1	Tarea 1	Tarea 1
Tarea 4	Tarea 4	Tarea 4

Tenemos que encontrar la manera de que las tareas pendientes pasen a mostrarse en pantalla ordenadas y tendría que mostrarse en pantalla de la siguiente manera:

Pendientes	En proceso	Finalizadas
Tarea 1	Tarea 2	Tarea 2
Tarea 2	Tarea 3	Tarea 3
Tarea 3	Tarea 1	Tarea 1
Tarea 4	Tarea 4	Tarea 4

MODIFICANDO EL .TXT

Para poder llevar a cabo el planteamiento anterior tenemos que asignarle a las tareas ya creadas del .txt un idº, en este caso vamos a sobrescribir las tareas ya asignadas en el .txt con un nº “;” y la tarea quedando de la siguiente manera (2;Tarea 2). En nuestro caso el .txt quedaría de la siguiente manera, teniendo en cuenta que no tienen que estar de manera ascendente, pero si tienen que mantener un orden correlativo:

2;Darle de comer al perro
 1;Comprar el pan
 3;Visitar a Juan
 4;Darle de comer al perro
 5;Comprar el pan
 7;Visitar a Juan
 8;Darle de comer al perro
 9;Comprar el pan
 6;Visitar a Juan
 0;Primera tarea pendiente

ORDENANDO POR ID EN APLICACIÓN WEB

Una vez ya tenemos unas ID asignadas a nuestras tareas, tenemos que hacer que estas tareas se muestren en pantalla y de manera ordenada en orden ascendente, en este caso sería (0,1,2,3,4,5,6,7,8,9).

Para poder realizar este proceso vamos a crear una variable llamada \$contador con valor 0 para llevar las cuentas de las tareas que se imprimen correctamente en el documento y decirle al programa que el primer ID que tiene que encontrar es el valor de \$contador.

Seguidamente vamos a utilizar un bucle while para ir leyendo todas las líneas del fichero. Después como nos interesa ordenar por id dentro del bucle while vamos a separar el contenido de cada línea en 2 variables con un list (\$id y \$tarea) e indicamos con un explode que queremos separar cuando llegue al “.”.

Una vez llegados a este punto solo nos faltaría comprobar con un if si el ID leído es igual que el ID que nosotros necesitamos que se imprima, en este caso nosotros al darle a \$contador un valor de 0 le indicamos que el ID que hay en el .txt tiene que ser igual que 0 ya que queremos que los ID comiencen por 0. Si el valor \$id no es igual que el valor de \$contador no entraría en el if, volvería a repetir el bucle while desde el comienzo leyendo la siguiente línea y asignándole a \$id el nuevo valor.

Este bucle se repetirá hasta que el valor \$id sea igual que el valor \$contador que en este caso cuando sean iguales va a imprimir en pantalla la id y la tarea correspondiente a ese id en este caso 0 y le añadiría un salto de línea,

le sumaria 1 a la variable contador para que la siguiente ID que busque ya no sea 0 si no 1 y volvería a leer desde el principio del fichero para no saltarse ningún ID con un fseek().

De esta manera una vez acabe todo el proceso vamos a obtener lo siguiente en pantalla:

Tareas Pendientes
0 .Primera tarea pendiente 1 .Comprar el pan 2 .Darle de comer al perro 3 .Visitar a Juan 4 .Darle de comer al perro 5 .Comprar el pan 6 .Visitar a Juan 7 .Visitar a Juan 8 .Darle de comer al perro 9 .Comprar el pan

```
<?php
/*Ejecutamos un bucle para organizar los id de manera secuencial ascendente
comenzando por el valor 0 del archivo pendientes.txt*/
$pendientes = fopen ('tareas/pendientes.txt', "rb");
$contador = 0;

while ($lineas =fgets ($pendientes)) {
    list($id, $tarea) = explode(";", $lineas);
    if($lineas == $contador){
        echo "$id .$tarea<br>";
        $contador++;
        fseek($pendientes,0);
    }
}
fclose ($pendientes);
?>
```

Etapa 6

INTRODUCCIÓN

En esta etapa vamos a realizar el siguiente requisito del proyecto:

- Crear las tareas **pendientes** desde la aplicación web

Para poder permitir que el cliente pueda añadir tareas al .txt sin tener que modificar el .txt manualmente hemos de realizar un formulario que se mostrará junto con las tareas pendientes para que pueda añadir tareas según necesite y hemos de realizar un archivo a parte donde tratar los datos que va a indicar el cliente y que esos datos que van a ser ID y tarea se escriban directamente en él .txt.

CREANDO UN FORMULARIO

Junto a las tareas que se muestran ordenadas en la Aplicación Web vamos a añadir un formulario con un <form> y vamos a indicar que el cliente pueda añadir por un <Input> el ID y por otro <Input> la tarea y el botón para enviar.

Nota¹: Intente hacer la tarea con un <textarea> pero la aplicación daba error ya que al poder introducir saltos de línea la aplicación no podía de ninguna manera leer el ID. La solución que encontré es poner un <input> para no dejar que el cliente pueda introducir saltos de línea (de esta manera evitamos que en el documento se encuentren huecos o saltos de línea indeseados) y que la aplicación tenga un correcto funcionamiento.

Una vez tenemos la estructura del formulario añadimos en action a donde queremos que vaya ese formulario en este caso nosotros lo vamos a enviar a “formproyectoetapa6.php” que es el archivo donde vamos a tratar estos datos y escribirlos en el fichero .txt.

Tareas Pendientes

Introducir Tarea Pendiente

ID:(Introducir nº ID secuencial)

Descripción Tarea

Introducir Tarea

0 . Primera tarea pendiente

1 . Comprar el pan

2 . Darle de comer al perro

3 . Visitar a Juan

4 . Darle de comer al perro

5 . Comprar el pan

6 . Visitar a Juan

7 . Visitar a Juan

8 . Darle de comer al perro

9 . Comprar el pan

```
<form method="POST" enctype="application/x-www-form-urlencoded" action="formproyectoetapa6.php">

  <fieldset>
    <legend>Introducir Tarea Pendiente</legend>
    <div> <label>ID:(Introducir nº ID secuencial) <br><input name="intid"> </label> </div>
    <div> <label>Descripción Tarea<br><input name="intpendientes"></label> </div>
  </fieldset>

  <div><button type="submit">Introducir Tarea</button></div>

</form>
```


TRATANDO LOS DATOS

Creemos un archivo .php para tratar los datos del formulario.

Lo primero que tenemos que hacer es abrir el fichero .txt ya que queremos introducir los datos dentro del fichero. Después vamos a guardar los datos introducidos por el cliente en el formulario en 2 variables, la ID en \$idform y la tarea en \$pendientesform.

Una vez tenemos guardados estos valores vamos a escribirlos en el fichero .txt con un fwrite() y cerramos el fichero ya que no lo vamos a utilizar más.

Y por último añadimos un enlace que vuelva a la Aplicación Web para comprobar si es número de ID introducido es correcto es decir continua con la secuencia del último ID se muestra en pantalla, Si introduce otro numero repetido no se va a mostrar ya que solo va a imprimir el que primero encuentre y si añade un número mayor hasta que el ID a imprimir no sea igual a ese ID no se va a mostrar.

 pendientes.txt: Bloc de notas

Archivo Edición Formato Ver Ayuda

```
2;Darle de comer al perro
1;Comprar el pan
3;Visitar a Juan
4;Darle de comer al perro
5;Comprar el pan
7;Visitar a Juan
8;Darle de comer al perro
9;Comprar el pan
6;Visitar a Juan
0;Primera tarea pendiente
10; Primera tarea por formulario
```



```
<?php

$pendientes = fopen ('tareas/pendientes.txt', "a+b");

#Guardamos el contenido del formulario en variables
$idform = $_POST["intid"];
$pendientesform = $_POST["intpendientes"];

/*Se escribe al final del archivo de pendientes.txt el ID y la tarea introducida por el usuario
no se realiza validación, si el número es menor o mayor no se muestra en pantalla pero no da ningún aviso*/
fwrite ($pendientes, " ".PHP_EOL . "$idform; $pendientesform");
fclose ($pendientes);

#Enlace para volver a la aplicación web y mostrar las tareas correctamente
echo '<a href="etapa6.php">Se ha añadido la tarea correctamente, pulsa aquí para continuar</a>';
```

Etapa 7

INTRODUCCIÓN

En esta etapa vamos a realizar los siguientes requisitos del proyecto:

- Cuando cree una nueva tarea pendiente, podrá especificar el id. Para ello:
 - El id que especifique no debe existir ya entre las tareas existentes
 - Si no especifica el id, se creará con el siguiente id que corresponda
- Para garantizar que los id's son correctos, se hará un validador que compruebe que los id's comienzan en 0, no se repite ninguno y no existen huecos.

VALIDADOR DE DATOS

Una vez el cliente puede introducir ID's y tareas tenemos que cerciorarnos que los id's sean correctos para que no pueda existir ningún tipo de fallo. Por ello tenemos que cambiar la forma en la que esta trabajando actualmente el fichero donde tratamos los datos del formulario.

Vamos a comenzar añadiendo un validador como en la Etapa 6 pero que no nos imprima en pantalla nada solo que verifique que los ID son correctos y cuando verifique el último ID la variable \$contador sepa que ID debería venir a continuación (Al estar verificada la secuencia el último valor que guarde \$contador será el ID correcto de manera inequívoca)

Nota²: El valor de ID será correcto salvo que el cliente modifique manualmente y erróneamente la secuencia de ID's el .txt (Se supone que actualmente el fichero .txt esta correctamente estructurado por ID's y tareas desordenadas, pero con un orden lógico y no pueden existir huecos, saltos de línea etc...)

Una vez sabemos el ID que tiene que introducirse a continuación, creamos un if donde si el id introducido en el formulario es diferente al id que debería tener la siguiente tarea nos imprime un mensaje de error, donde se indica el ID que introdujo el cliente , el ID que debería haber introducido y que se va a modificar automáticamente el valor introducido por el cliente por el valor correcto y cambiamos el valor de la variable del ID introducida por el cliente por el valor de contador.

Realizando el paso anterior nos aseguramos de que si el ID es incorrecto se modifique por el valor correcto y en el caso de ser correcto (al ser correcto) no cambia nada.

Como último paso del validador escribimos en el fichero la ID correcta y la tarea.

Nota³: Hay que destacar que el cliente no tiene que poner el “;” en el campo de ID ya que al escribir en el fichero nosotros indicamos que se escriba el ID y seguidamente el “;”. De esta manera evitamos que el cliente pueda equivocarse al poner o no “;” y no obtener un correcto funcionamiento de la aplicación.

Suponiendo que el cliente no modifica manualmente el .txt erróneamente con este sistema de validación solucionamos todos los requisitos del proyecto indicados anteriormente ya que:

- ✓ Si existe le asigna el valor que debería tener real.
- ✓ Si no especifica el ID se le asigna el valor que debería tener real.
- ✓ Se realiza un validador que comprueba que el ID comienza en 0.
- ✓ Al introducir la opción <input> en el formulario evitamos que el cliente pueda introducir saltos de línea indeseados que puedan interferir en la correcta ejecución de la Aplicación Web.

← → ↻ ⓘ localhost/repositorio/Proyecto1/formproyecto.php

Cuidado! El valor del ID introducido (2345563) tendria que ser (11)
 No te preocupes ya hemos realizado el cambio por ti :)
[Se ha introducido la tarea correctamente, pulsa aquí para continuar](#)

Tareas Pendientes

Introducir Tarea Pendiente

ID:(Introducir nº ID secuencial)

Descripción Tarea

0 . Primera tarea pendiente

1 . Comprar el pan

2 . Darle de comer al perro

3 . Visitar a Juan

4 . Darle de comer al perro

5 . Comprar el pan

6 . Visitar a Juan

7 . Visitar a Juan

8 . Darle de comer al perro

9 . Comprar el pan

10 . Primera tarea por formulario

11 . Segunda tarea por formulario validada

```
<?php

$pendientes = fopen ('tareas/pendientes.txt', "a+b");

#Guardamos el contenido del formulario en variables
$contador = 0;
$idform = $_POST["intid"];
$pendientesform = $_POST["intpendientes"];

/*Ejecutamos un bucle para organizar los id de manera secuencial ascendente
comenzando por el valor 0 del archivo pendientes.txt */
while ($lineas =fgets ($pendientes)){

    list($id, $tarea) = explode(";", $lineas);

    if($lineas == $contador){

        $contador++;
        fseek($pendientes,0);

    }

}

/*Ejecutamos un bucle para validar que el nº de ID introducido sea correcto
de no ser correcto automáticamente se le asigna el valor correcto que debería tener
y avisa al usuario de los cambios producidos*/

if ($idform != $contador){

    echo "Cuidado! El valor del ID introducido ($idform) tendria que ser ($contador)<br>No te preocupes ya hemos realizado el cambio por ti :)<br>";
    $idform = $contador;

}

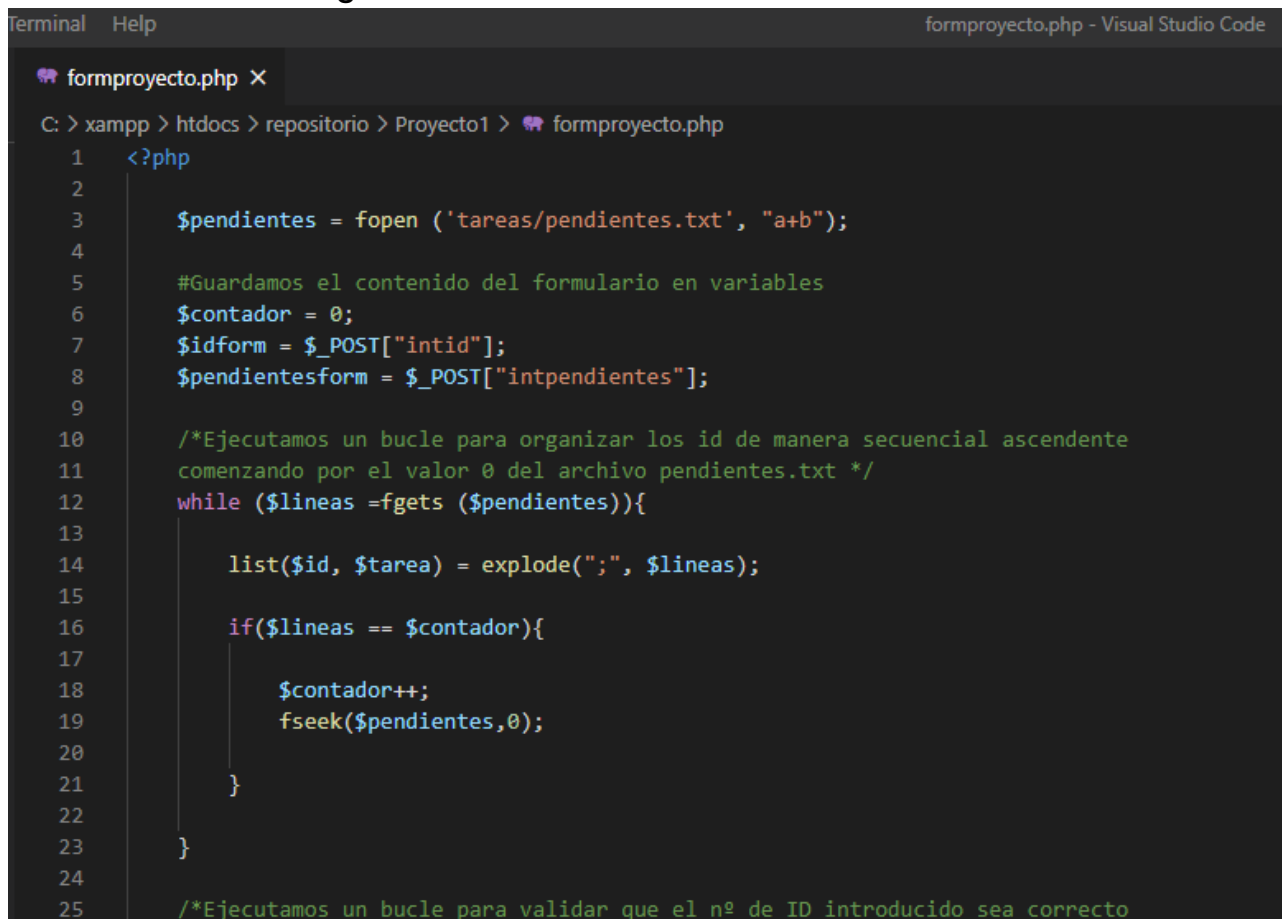
#Se escribe en el documento pendientes.txt la ID correcta y su tarea
fwrite ($pendientes, " ".PHP_EOL . "$idform; $pendientesform");
fclose ($pendientes);

#Enlace para volver a la aplicación web y mostrar las tareas correctamente
echo '<a href="etapa7.php">Se ha introducido la tarea correctamente, pulsa aquí para continuar</a>';
```

Herramientas utilizadas

EDITOR DE CÓDIGO

Para escribir el código he utilizado el editor Visual Estudio code.

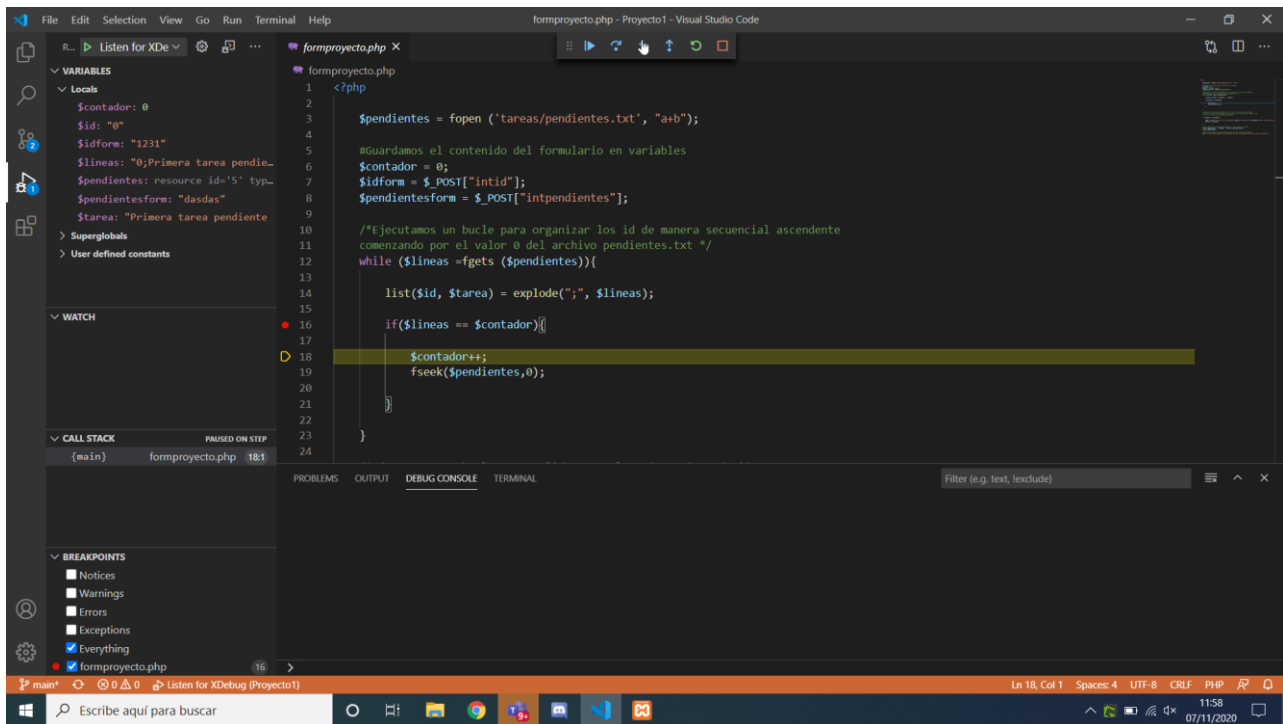


```

formproyecto.php X
C: > xampp > htdocs > repositorio > Proyecto1 > formproyecto.php
1  <?php
2
3      $pendientes = fopen ('tareas/pendientes.txt', "a+b");
4
5      #Guardamos el contenido del formulario en variables
6      $contador = 0;
7      $idform = $_POST["intid"];
8      $pendientesform = $_POST["intpendientes"];
9
10     /*Ejecutamos un bucle para organizar los id de manera secuencial ascendente
11     comenzando por el valor 0 del archivo pendientes.txt */
12     while ($lineas =fgets ($pendientes)){
13
14         list($id, $tarea) = explode(";", $lineas);
15
16         if($lineas == $contador){
17
18             $contador++;
19             fseek($pendientes,0);
20
21         }
22     }
23
24
25     /*Ejecutamos un bucle para validar que el nº de ID introducido sea correcto
  
```

DEPURADOR DE CÓDIGO

Para escribir el código he utilizado el XDebug el editor Visual Estudio code. Adjunto captura de pantalla donde se comprueba que si el valor del id de la línea es 0 entra en el bucle y lo imprime en pantalla y si no continúa buscando hasta encontrarlo.



Conclusión

PROS DE LOS FICHEROS SECUENCIALES

Son rápidos si se accede secuencialmente.

Utilizan mejor el espacio y son sencillos de usar y aplicar.

Se pueden abrir como lectura o escritura o los dos.

Al no poder dejar huecos vacíos se almacenan muchos más datos.

CONTRAS DE LOS FICHEROS SECUENCIALES

Son muy lentos si los datos a los que acceder no se encuentran secuencialmente.

Para acceder a un registro tienes que pasar por todos los anteriores tardando el doble de lo normal en acceder a el.

No permite la utilización de varios usuarios simultáneos.

Solo lee hacia adelante

La estructura de los campos es muy rígida