

Curso: SEPE10-22: Desarrollo web BackEnd

Docente: David Alcolea

PLA 2_ DETALLE Y DESCRIPCIÓN

Nombre del PLA: Formulario para calcular el coste de estancia en un hotel usando funciones

Objetivos del PLA:

- Trabajar con funciones del lenguaje php
- Permanencia de datos en un formulario
- Uso de constantes
- Tratamiento de errores
- Analizar documentación de requerimientos de usuario

Competencias asociadas al PLA:

Competencias técnicas	Soft Skills
<ul style="list-style-type: none">• Uso de funciones• Formularios síncronos con permanencia de datos• Tratamiento de errores con try...catch	<ul style="list-style-type: none">• Resolución de problemas• Interpretar requerimientos• Gestionar un proyecto• Búsqueda, gestión y uso de la información

Calendario y horas de dedicación:

Del 10/10/22 al 11/10/22

Síncronas	Webinars	3
	Tutorías presenciales	0
Asíncronas		7
Presenciales		
Totales		10

Modalidad PLA: Ejercicio independiente

Instrucciones que recibirá el alumno para resolver el reto:

En este reto veremos el uso de funciones en el lenguaje php y profundizaremos un poco más en el tratamiento de errores

Instrucciones metodológicas para la entrega de los retos:

Este PLA consta de una actividad consistente en la entrega de un ejercicio en donde capturaremos unos datos de un formulario, que se facilitará como recursos del PLA, y, utilizando funciones, se realizará el cálculo de un precio que mostraremos en el mismo formulario.

PLA02: COSTE HOTEL

Número de noches:

Destino:

Días alquiler coche:

Coste total:

El formato de entrega de la actividad es el siguiente:

Pla2_tu_nombre_tu_apellido.zip

La entrega del fichero será a través de la aplicación Moodle del curso.

Las actividades son individuales y intransferibles.

La resolución de dudas o comentarios se podrán realizar a través de las herramientas de comunicación de la aplicación Moodle del curso

La actividad se entregará antes de la fecha límite de entrega.

Requerimientos de usuario (PLA 02: Funciones php)

Tendremos que desarrollar un pequeño formulario para calcular el precio de un viaje en función de los parámetros informados en el formulario adjunto

PLA02: COSTE HOTEL

Número de noches:

Destino:

Días alquiler coche:

Coste total:

Especificaciones del usuario:

- Se calculará el precio del hotel a partir del número de noches de hotel del formulario (a razón de 60 € por noche)

EL número mínimo de noches es de 1

- Se escogerá una ciudad de destino de entre las disponibles en el desplegable y se calculará el precio del viaje de avión:

Madrid	: 150 €
París	: 250 €
Los Angeles	: 450 €
Roma	: 200 €

No se permitirá ninguna otra ciudad que las anteriormente indicadas

- Opcionalmente, se informarán los días de alquiler de coche y se calculará el precio del mismo teniendo en cuenta los siguientes precios:
 - Cada día de alquiler cuesta 40 €
 - Si se alquila por tres o más días se obtiene un descuento de 20 €.
 - Si se alquila por siete días o más se obtiene un descuento de 50 € (no acumulable con los 20 € de haber alquilado por más de 3 días)

No se permitirá que el número de días de alquiler supere el número de días de estancia en el hotel

- Al pulsar 'enviar' se mostrará el precio final en el área específica del formulario
- Cualquier incidencia o error de validación se mostrará en el formulario

Requerimientos técnicos solicitados

- Uso de funciones para el cálculo de los precios
- Uso de constantes para los valores fijos
- Los datos del formulario deben permanecer tanto en caso de cálculo correcto como el caso de que se produzca algún error
- Se recomienda el uso de `try...catch` para el tratamiento de errores

Explicación Técnica del Reto (PLA 02: Funciones php)

EJERCICIO 1: FORMULARIO DE CAPTURA DE DATOS

Crear una página llamada **PLA02_Coste_Hotel.php** con las especificaciones que se enumeran a continuación:

PLA02: COSTE HOTEL

Número de noches:

Destino:

Días alquiler coche:

Coste total:

1.- Características del formulario

El formulario se entrega ya confeccionado dentro del fichero **PLA2_recursos.zip** pero podéis confeccionar el vuestro propio

- Control de tipo number para captura del número de noches
- Combo para la selección del destino:
 - Madrid
 - París
 - Los Angeles
 - Roma
- Control de tipo number para captura del número de días de alquiler de coche (opcional)
- Control de tipo submit para el envío del formulario al servidor
- Control de tipo texto y desactivado (atributo `disabled`) para mostrar el texto con el precio calculado del viaje
- Una etiqueta `` o similar para mostrar los posibles errores de validación de datos

NOTA: A diferencia del PLA anterior, en donde el tratamiento del formulario lo realizábamos en un fichero distinto, en este caso vamos a utilizar el mismo fichero donde capturamos los datos para realizar las validaciones y el cálculo del precio. Para ello en el atributo `action` de la etiqueta `form` indicamos #:

```
<form method="post" action="#">
```

EJERCICIO 2: CALCULO DEL PRECIO DEL HOTEL A PARTIR DEL NUMERO DE NOCHES

Vamos a ver el procedimiento que tendremos que implementar para el cálculo del precio del hotel:

Número de noches:

4

1.- Comprobar cuando se pulsa el botón de submit

PHP, a diferencia del lenguaje JavaScript, no tiene eventos que podamos utilizar para saber cuando el usuario pulsa un botón o realiza alguna acción sobre un formulario. Tendremos que detectar la pulsación del botón de forma indirecta a partir de los atributos `name` de los controles del formulario y que llegarán al servidor dentro de la súper variable `$_POST`.

Veamos un ejemplo: una vez completado el formulario, pulsamos el botón de enviar y miramos el contenido que llega al servidor en la variable `$_POST`. Para ello colocamos como primera línea de código php la siguiente:

```
print_r($_POST);
```

NOTA: `print_r()` se utiliza para visualizar el contenido de un array (los veremos en detalle en el siguiente PLA)

Al entrar en la página por primera vez podemos ver lo siguiente:

```
array()
```

Es decir, como no hemos pulsado el botón de `submit` del formulario el array aparece vacío

Ahora vamos a informar algunos datos, por ejemplo número de noches y ciudad y pulsamos el botón de enviar. Veremos:

```
Array ( [noches] => 4 [ciudad] => Paris [coche] => 0 [enviar] => Enviar )
```

Vemos como las claves asociativas del `array` (en negrita) coinciden exactamente con el valor que hemos dado a los atributos `name` del formulario

Parece obvio que el código que tenemos que añadir para calcular el precio del hotel solo se tiene que ejecutar cuando el usuario pulsa el botón y no al entrar a la página por primera vez. Para ello utilizaremos una función nativa de php que nos indicará si una variable o el índice de un array existe y, de ser así, ejecutamos el código que se encargue de recoger los datos y calcular el precio. Vamos a detectar cuando nos llega el índice que corresponde al atributo `name` de botón:

```
if (isset($_POST['enviar'])) {  
    //aquí pondremos el código necesario para recoger los datos, validarlos y  
    realizar el cálculo del precio  
}
```

2.- Especificaciones cálculo del precio

Para calcular el precio del hotel realizaremos los siguientes pasos:

- Recuperar el dato correspondiente al número de noches (utilizar `$_POST['noches']` o `filter_input()` indistintamente)

- Validar que el número de noches sea numérico y mayor que cero. En caso contrario mostraremos en el área de errores del formulario (``) un mensaje de error del tipo “Noches debe ser numérico y mayor que 0” o similar.

NOTA: Si utilizamos `filter_input()` con el filtro `FILTER_VALIDATE_INT` ya se validará automáticamente que el contenido sea numérico. Si recogemos el dato con `$_POST` para asignarlo a una variable podemos validar posteriormente que sea numérico con:

```
if (!is_numeric($noches)) {...}
```

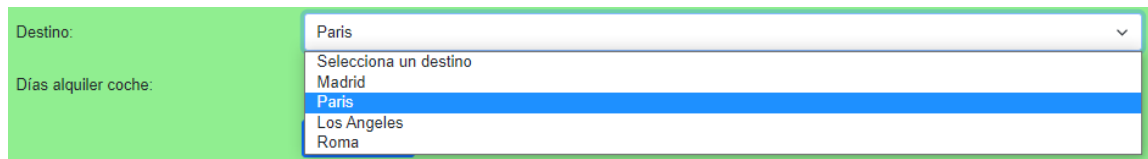
- Creamos una función llamada `costeHotel()` o similar con un parámetro de entrada que será el número de noches capturado del formulario. La función devolverá el precio del hotel considerando que cada noche cuesta 60 €.

El resultado del cálculo lo tendremos que recoger en la línea de código donde llamamos a la función.

NOTA: Las funciones las podéis construir al principio o al final del programa. En este caso por claridad de código se recomienda al final pero OJO, siempre fuera del `if` con el `isset()` que hemos utilizado para detectar cuando el usuario pulsa el botón

EJERCICIO 3: CALCULO DEL PRECIO DEL AVION A PARTIR DE LA CIUDAD SELECCIONADA

Vamos a ver el procedimiento que tendremos que implementar para el cálculo del precio del avión:



1.- Especificaciones cálculo del precio

Para calcular el precio del avión realizaremos los siguientes pasos:

- Recuperar el dato correspondiente a la ciudad seleccionada de la combo
- Validar que se haya seleccionado una ciudad. En caso contrario mostrar un error del tipo “Se debe seleccionar una ciudad” o similar

NOTA: De forma general podemos validar si una variable tiene o no contenido utilizando la siguiente expresión:

```
if (empty($ciudad)) { ... }
```

- Creamos una función llamada `costeAvion()` o similar con un parámetro de entrada que será el nombre de la ciudad capturada del formulario. La función devolverá el precio del avión considerando la siguiente tabla de precios:

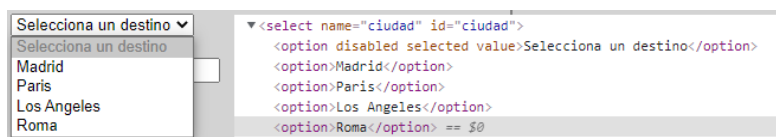
Madrid	: 150 €
París	: 250 €
Los Angeles	: 450 €
Roma	: 200 €

- Si la ciudad seleccionada no corresponde con ninguna de las anteriores mostraremos un error del tipo “Ciudad no válida” o similar

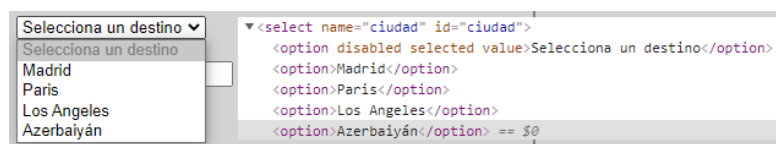
NOTA: ¿Es posible que al servidor llegue una ciudad que, inicialmente, no se encuentre en la combo cuando cargamos la página? Pues la respuesta es que SI, de aquí la importancia de validar todas las casuísticas posibles en el servidor.

Imaginemos un usuario con conocimientos medios o avanzados de HTML: Si en el navegador pulsamos F12 para abrir la consola vemos que podemos editar cualquier opción de la combo o cambiar valores de otros controles del formulario solo pulsado sobre ellos:

Inicialmente tenemos:



Pero si pulsamos sobre la opción *Roma* en la consola podemos cambiar su contenido:



Hemos añadido un destino que no tenemos contemplado en el servidor como ciudad válida para realizar el cálculo del precio

- El resultado del cálculo lo tendremos que recoger en la línea de código donde llamamos a la función.

EJERCICIO 4: CALCULO DEL PRECIO DEL COCHE A PARTIR DE LOS DIAS DE ALQUILER

Vamos a ver el procedimiento que tendremos que implementar para el cálculo del precio del alquiler de coche (en caso que se haya informado):

Días alquiler coche:	2
----------------------	---

1.- Especificaciones cálculo del precio

Para calcular el precio del alquiler de coche realizaremos los siguientes pasos:

- Recuperar el dato correspondiente a los días de alquiler
- En caso que se haya informado algún valor, validar que el contenido sea numérico y mayor o igual a cero. Mostrar un mensaje de error del tipo “Días de alquiler debe ser mayor o igual a 0” en caso contrario.
- Validar que el número de días de alquiler no supere el número de días de estancia en el hotel. Mostrar un mensaje similar a “Días de alquiler no puede superar a días de estancia” en caso contrario.
- Creamos una función llamada `costeCoche()` o similar con un parámetro de entrada que será el número de días de alquiler capturados del formulario. La función devolverá el precio del coche considerando la siguiente tabla de precios:
 - Cada día de alquiler cuesta 40 €
 - Si se alquila por tres o más días se obtiene un descuento de 20 €.

- Si se alquila por siete días o más se obtiene un descuento de 50 € (no acumulable con los 20 € de haber alquilado por más de 3 días)
- El resultado del cálculo lo tendremos que recoger en la línea de código donde llamamos a la función.

EJERCICIO 5: CALCULO TOTAL DEL PRECIO DEL VIAJE

Por último, a partir de los tres precios anteriores, calcularemos el precio total del viaje que mostraremos en el `input` habilitado al efecto

Formulario de cálculo de coste de hotel (PLA02: COSTE HOTEL) con los siguientes campos:

- Número de noches:
- Destino:
- Días alquiler coche:
- Botón: Enviar datos
- Coste total:

EJERCICIO 6: PERMANENCIA DE DATOS EN EL FORMULARIO

Si hemos llegado hasta aquí, y hemos probado el ejercicio, es posible que hayamos detectado que, cuando se produce un error de validación, perdemos los datos introducidos en el formulario de forma que obligamos al usuario a volver a teclearlos.

Esto es una consecuencia de trabajar con formularios síncronos, es decir, formularios que vuelven a cargarse de nuevo cada vez que pulsamos el botón de enviar y el servidor recarga la página.

Para evitar este inconveniente hoy en día se utiliza la tecnología AJAX, pero, de momento vamos a solucionarlo utilizando únicamente código php.

NOTA: Esto es un tema básico de usabilidad, imaginad que en todos los formularios que cumplimentamos en nuestro día a día tuviéramos que reintroducir todos los datos si se produce un error de validación

1.- Conservar los datos de los controles de tipo `<input>`

Para los controles de noches de hotel y días de alquiler de coche tan solo tenemos que indicar en las correspondientes etiquetas `input` un atributo `value` con el valor que queremos informar cada vez que se recarga la página:

- Tendrán que aparecer vacíos cuando entremos por primera vez a la página
- Tendrá que aparecer el valor que ha informado el usuario antes de pulsar el botón una vez el servidor recargue la página después de realizar el cálculo del precio o se produzca algún error de validación.

Vemos un ejemplo de como quedaría nuestra etiqueta `input` para número de noches:


```
<input type="number" name="noches" id="noches" value='<?=$noches;?>'>
```

OJO: Recordad que la variable php `$noches` tiene que existir cuando el servidor envíe la página al navegador la primera vez que entramos

2.- Conservar los datos de los controles de tipo `<option>`

Para el control de selección de ciudad no podemos utilizar la técnica descrita en el apartado anterior. En este caso necesitamos que la etiqueta `option` que corresponda a la ciudad que ha seleccionado el usuario contenga el atributo `selected`.

Si ninguna de ellas tiene el atributo `selected` aparecerá seleccionada la primera de ellas (la opción 'selecciona un destino')

Para añadir el atributo `selected` a la `option` correspondiente solo tendremos que preguntar dentro de la propia etiqueta `option` con un tag de apertura y cierre de php, si el contenido de la etiqueta corresponde con el contenido de la variable utilizada para capturar el dato ciudad en el servidor y, de ser así, añadir el atributo `selected` utilizando una instrucción `echo`. Veamos un ejemplo

```
<option <?php if ($ciudad == 'Madrid') {echo 'selected';} ?> >Madrid</option>
```

EJERCICIO 7: USO DE CONSTANTES

Vamos a optimizar un poco más el ejercicio para utilizar constantes para aquellos valores que son fijos y no dependen de los datos que introduce el usuario en el formulario:

- Precio de la noche de hotel
- Precio del día de alquiler de coche
- Descuento para 3 días de alquiler
- Descuento para 7 días de alquiler

En este caso podemos crear constantes que podemos utilizar en todo el ámbito del programa incluso dentro de las funciones sin necesidad de pasarlas como parámetro ya que, en PHP, todas las constantes tienen visibilidad en todo el código (son globales). Por ejemplo, nos podríamos crear una constante al principio del programa para el precio de la noche de hotel:

```
const PRECIO_HOTEL = 60;
```

EJERCICIO 8: CONTROL DE ERRORES CON TRY...CATCH

Por último, si no lo habéis incluido ya, vamos a refactorizar el ejercicio (cambiar el código) para añadir un control más eficiente de errores utilizando el sistema `try...catch`.

Este sistema tiene una ventaja enorme cuando tenemos que validar varios datos en secuencia ya que separa el flujo del programa en dos vías:

- El flujo principal que contiene la lógica de negocio en caso de que no se produzcan errores
- El flujo a ejecutar en caso que se produzca algún error

El resultado es un código más limpio y fácil de mantener. Su funcionamiento es el siguiente:

1. Todas las validaciones de datos y, MUY IMPORTANTE, el código que depende de estas validaciones (el código que se va a ejecutar en caso que no se produzcan errores) se colocará en un bloque `try { ... }`
2. Si al realizar una validación se produce un error (por ejemplo el número de noches no es numérico) lanzaremos una excepción de una clase específica de php llamada **Exception** indicando el mensaje de error que queremos asociar a esta validación y, opcionalmente, un código de error (cuando veamos programación orientada a objetos entenderemos mejor los conceptos de clases y métodos)

```
if (!is_numeric($noches)) {  
    throw new Exception("Noches debe ser numérico", 10);  
}
```

3. Inmediatamente después del bloque `try { ... }` debe existir un bloque `catch { ... }` para poder capturar las excepciones lanzadas y procesarlas.

```
try {  
    //validar datos  
    //lógica de negocio si no hay errores  
} catch (Exception $error) {  
    //método del objeto $error de la clase Exception para recuperar el mensaje  
    de error  
    $mensaje = $error->getMessage();  
    //método del objeto $error de la clase Exception para recuperar el código  
    de error  
    $codigo = $error->getCode();  
}
```

Recursos de aprendizaje vinculados:

Toda la documentación (manuales, vídeo tutoriales, esquemas, etc) se encontrará en el aula virtual así como los recursos necesarios para resolver el reto (imágenes, etc)

- Manual de funciones en PHP