Teoría del Método GET en PHP

Cuando un usuario interactúa con una página web, es común que desee enviar información al servidor para ser procesada. Esto se logra generalmente a través de formularios web, y en PHP, los datos de un formulario se pueden enviar utilizando los métodos **GET** o **POST**. En esta sección, nos enfocaremos en cómo funciona el **método GET**.

¿Qué es el Método GET?

El método GET es uno de los dos métodos principales para enviar datos de un formulario a un servidor. En GET, los datos del formulario se envían como parte de la URL de la solicitud HTTP. Esto significa que los valores ingresados por el usuario en el formulario se anexan a la URL en formato de **cadena de consulta**.

Un ejemplo de una URL que contiene datos enviados con GET sería:

```
http://mi-sitio.com/procesar.php?nombre=Juan&edad=25
```

En este caso, nombre y edad son las **claves** (nombres de los campos del formulario) y Juan y 25 son los **valores** que el usuario ha ingresado en esos campos.

Características del Método GET

1. **Visibilidad de los datos en la URL**: Los datos enviados con GET se añaden a la URL, lo que los hace visibles para el usuario y cualquier persona con acceso al enlace.
2. **Limitaciones de tamaño**: Las URL tienen un límite de longitud (alrededor de 2048 caracteres en la mayoría de los navegadores), por lo que no es ideal para enviar grandes cantidades de datos.
3. **Datos no sensibles**: No es seguro para enviar datos sensibles (como contraseñas), ya que los valores se muestran en la URL.
4. **Cacheabilidad**: Las solicitudes GET pueden ser cacheadas por los navegadores, lo que es útil para operaciones como búsqueda o filtros.
5. **Facilidad de compartir**: Como los datos son parte de la URL, el enlace se puede compartir fácilmente y los resultados se mantendrán consistentes.

¿Cómo funcionan los formularios con GET?

Cuando se utiliza el método GET en un formulario HTML, los datos ingresados por el usuario se envían al servidor en la URL. En el lado del servidor, PHP puede acceder a estos datos mediante la variable superglobal **$_GET**, que es un array asociativo que contiene los datos enviados.

Ejemplo de Formulario con GET

HTML (Formulario):

```html
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <title>Formulario con GET</title>
</head>
<body>
    <h1>Enviar Datos con GET</h1>
    <form action="process.php" method="get">
        <label for="name">Name:</label>
        <input type="text" name="name" required>
        <br><br>
        <label for="age">Age:</label>
        <input type="number" name="age" required>
        <br><br>
        <input type="submit" value="Submit">
    </form>
</body>
</html>
```

PHP (process.php):

```php
<?php
// Recoger los datos enviados con GET
$name = $_GET['name'];
$age = $_GET['age'];

// Mostrar los datos
echo "Name: " . $name . "<br>";
echo "Age: " . $age . " years old";
?>
```

Al enviar este formulario, los datos ingresados en los campos de "name" y "age" se envían a la URL como:

`http://mi-sitio.com/process.php?name=John&age=25`

PHP recoge estos valores usando $_GET y puede mostrarlos o procesarlos.

Ejercicio 1: Acceso VIP en un Festival de Música

Teoría: Condicionales en PHP

Un condicional **if** en PHP permite ejecutar un bloque de código solo si se cumple una condición específica. La estructura básica de un condicional es:

```
if (condición) {
    // Código que se ejecuta si la condición es verdadera
}
```

Cuando el valor en la condición es verdadero, el bloque de código se ejecuta; si no lo es, el bloque no se ejecuta. Puedes usar también **else** para ejecutar código si la condición no es verdadera.

Ejemplo simple de if en PHP

```
<?php
$amountSpent = 150;

if ($amountSpent > 100) {
    echo "Congratulations! You have unlocked VIP access.";
}
```

En este ejemplo, si la variable $amountSpent es mayor que 100, el usuario recibe un mensaje de que ha desbloqueado el acceso VIP.

Ejemplo Completo: Verificación de Edad para Acceso

Imagina que estás creando un formulario donde los usuarios ingresan su edad para determinar si tienen acceso a ciertas funciones restringidas para mayores de 18 años. Usaremos GET para enviar la información.

Código HTML (formulario):

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <title>Verificación de Edad</title>
</head>
<body>
    <h1>Verificación de Edad</h1>
    <form action="verify_age.php" method="get">
        <label for="age">Age:</label>
        <input type="number" name="age" required>
        <br><br>
        <input type="submit" value="Verify">
    </form>
</body>
</html>
```

Código PHP (verify_age.php):

```php
<?php
// Recogemos la edad ingresada por el usuario
$age = $_GET['age'];

// Verificamos si el usuario tiene al menos 18 años
if ($age >= 18) {
    echo "Welcome! You are allowed to access the content.";
} else {
    echo "Sorry, you must be at least 18 years old to access this content.";
}
?>
```

Enunciado del Ejercicio 1: Verificación de Acceso VIP para un Festival de Música

Estás a cargo de crear un sistema para un **festival de música**, donde los asistentes pueden obtener acceso VIP si han gastado más de **300 euros** en entradas. Tu objetivo es desarrollar un sistema de verificación que determine si un usuario ha gastado lo suficiente para desbloquear el acceso VIP.

Instrucciones:

1. Crea un formulario en HTML donde los usuarios puedan ingresar el **total gastado** en sus entradas.
2. Usa el método **GET** para enviar los datos al servidor.
3. En el servidor (archivo PHP), recoge el valor total usando $_GET.
4. Usa un condicional **if** para verificar si el total es mayor de **300 euros**.
5. Si el total es mayor de 300 euros, muestra un mensaje que diga **"¡Enhorabuena! Has desbloqueado acceso VIP"**.
6. Si el total es menor o igual a 300 euros, muestra un mensaje de **"Lo sentimos, no calificas para acceso VIP"**.

Ejercicio 2: Formulario de Registro para un Torneo de eSports

Teoría: Condicionales if...else en PHP

El condicional **if...else** permite ejecutar un bloque de código si la condición es verdadera y otro bloque de código diferente si la condición es falsa. Esto es útil para manejar situaciones donde hay solo dos resultados posibles.

La estructura básica de **if...else** es:

```
if (condición) {
    // Código si la condición es verdadera
} else {
    // Código si la condición es falsa
}
```

Ejemplo de if...else:

```
$playerLevel = 15;

if ($playerLevel >= 10) {
    echo "You are eligible to participate in the tournament!";
} else {
    echo "Sorry, you must be at least level 10 to enter the tournament.";
}
```

Ejemplo Completo: Verificación de Nivel para un Torneo de Videojuegos

Imagina que estás organizando un **torneo de videojuegos**. Solo los jugadores que han alcanzado el nivel 10 pueden inscribirse. Recogeremos el nivel del jugador y determinaremos si es elegible para participar.

Código HTML (formulario):

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <title>Verificación de Nivel</title>
</head>
<body>
    <h1>Verificación de Nivel para el Torneo</h1>
    <form action="verify_level.php" method="get">
        <label for="level">Player Level:</label>
```

```
        <input type="number" name="level" required>
        <br><br>
        <input type="submit" value="Check Eligibility">
    </form>
</body>
</html>
```

Código PHP (verify_level.php):

```php
<?php
// Recogemos el nivel del jugador
$level = $_GET['level'];

// Verificamos si el nivel es suficiente para participar en el torneo
if ($level >= 10) {
    echo "You are eligible to participate in the tournament!";
} else {
    echo "Sorry, you must be at least level 10 to enter the tournament.";
}
?>
```

Enunciado del Ejercicio 2: Registro en Torneo de eSports

Estás desarrollando una plataforma de inscripción para un torneo de **eSports**. Solo los jugadores que han alcanzado **al menos el nivel 15** pueden registrarse. Tu tarea es crear un formulario que recoja el nivel del jugador y determine si puede participar en el torneo.

Instrucciones:

1. Crea un formulario en HTML donde el jugador ingrese su **nivel actual**.
2. Usa el método **GET** para enviar los datos al servidor.
3. En el servidor (archivo PHP), recoge el valor del nivel usando $_GET.
4. Usa un condicional **if...else** para verificar si el jugador tiene **nivel 15 o superior**.
5. Si el jugador tiene nivel suficiente, muestra un mensaje de **"Puedes participar en el torneo"**.
6. Si el jugador no tiene el nivel requerido, muestra un mensaje de **"Lo sentimos, no tienes el nivel suficiente para participar"**.

Ejercicio 3: Sistema de Evaluación de una Competición de Diseño Gráfico

Teoría: El Uso del Condicional if...else if...else en PHP

A veces, necesitamos más de dos condiciones para evaluar diferentes escenarios en nuestro código. En estos casos, podemos usar **if...else if...else**. Este condicional permite evaluar múltiples condiciones, ejecutando diferentes bloques de código dependiendo del valor de una variable o de una condición.

La estructura básica de **if...else if...else** es:

```
if (condición 1) {
    // Código que se ejecuta si la condición 1 es verdadera
} elseif (condición 2) {
    // Código que se ejecuta si la condición 2 es verdadera
} else {
    // Código que se ejecuta si ninguna de las condiciones anteriores es
verdadera
}
```

Ejemplo simple de if...else if...else:

```php
<?php
$score = 85;

if ($score >= 90) {
    echo "You received an A!";
} elseif ($score >= 80) {
    echo "You received a B!";
} else {
    echo "You need to study more!";
}
?>
```

En este ejemplo, dependiendo de la puntuación del usuario, se mostrará un mensaje diferente.

Ejemplo Completo: Asignación de Prioridades a Tareas en un Proyecto de Desarrollo Web

Imagina que estás gestionando un proyecto de desarrollo web y necesitas asignar **niveles de prioridad** a diferentes tareas según su complejidad. Los niveles de dificultad van del 1 al 3, y dependiendo de la dificultad, asignas una prioridad.

Código HTML (formulario):

```html
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <title>Asignación de Prioridad</title>
</head>
<body>
    <h1>Asignación de Prioridad a Tareas</h1>
    <form action="assign_priority.php" method="get">
        <label for="difficulty">Difficulty Level:</label>
        <select name="difficulty" required>
            <option value="1">Level 1</option>
            <option value="2">Level 2</option>
            <option value="3">Level 3</option>
        </select>
        <br><br>
        <input type="submit" value="Assign Priority">
    </form>
</body>
</html>
```

Código PHP (assign_priority.php):

```php
<?php
// Recogemos el nivel de dificultad de la tarea
$difficulty = $_GET['difficulty'];

// Asignamos una prioridad basada en el nivel de dificultad
if ($difficulty == 1) {
    echo "The task has been assigned a low priority.";
} elseif ($difficulty == 2) {
    echo "The task has been assigned a medium priority.";
} else {
    echo "The task has been assigned a high priority.";
}
?>
```

Enunciado del Ejercicio 3: Evaluación de una Competición de Diseño Gráfico

Eres el encargado de un **concurso de diseño gráfico** y necesitas evaluar las obras de los participantes en base a las calificaciones otorgadas por los jueces. Dependiendo de la **calificación final** (que va del 1 al 100), debes asignarles una medalla: Oro, Plata o Bronce. Los participantes que no lleguen a la puntuación mínima no recibirán una medalla.

Instrucciones:

1. Crea un formulario en HTML donde los jueces puedan ingresar la **calificación final** de cada participante.
2. Usa el método **GET** para enviar los datos al servidor.
3. En el servidor (archivo PHP), recoge la calificación usando $_GET.
4. Usa un condicional **if...else if...else** para determinar:
    - Si la calificación es **mayor o igual a 90**, asigna una **Medalla de Oro**.
    - Si la calificación es **mayor o igual a 75**, asigna una **Medalla de Plata**.
    - Si la calificación es **mayor o igual a 60**, asigna una **Medalla de Bronce**.
    - Si la calificación es **menor a 60**, muestra un mensaje de **"No se ha asignado medalla."**

Ejercicio 4: Horarios de Apertura de una Tienda Virtual

Teoría: El Uso del switch en PHP

El **switch** es otra estructura de control condicional en PHP, similar a if...else if, pero más limpia y fácil de leer cuando se tienen muchas condiciones que dependen del valor de una sola variable. La declaración switch evalúa una expresión y compara su valor con diferentes casos.

La estructura básica de un **switch** es:

```php
<?php
switch (variable) {
    case valor1:
        // Código si variable es igual a valor1
        break;
    case valor2:
        // Código si variable es igual a valor2
        break;
    default:
        // Código si ninguna de las anteriores coincide
}
```

Ejemplo simple de switch:

```php
<?php
$dayOfWeek = 3;

switch ($dayOfWeek) {
    case 1:
        echo "Today is Monday";
        break;
    case 2:
        echo "Today is Tuesday";
        break;
    case 3:
        echo "Today is Wednesday";
        break;
    default:
        echo "Invalid day";
}
?>
```

En este ejemplo, el valor de la variable $dayOfWeek se compara con cada uno de los casos. Si el valor es igual a 3, se muestra "Today is Wednesday".

Ejemplo Completo: Horarios de Apertura de una Tienda Online

Imagina que estás creando un sitio web para una tienda online que tiene diferentes **horarios de apertura** según el día de la semana. Queremos permitir que los usuarios seleccionen el día y luego mostrarles los horarios de apertura correspondientes.

Código HTML (formulario):

```html
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <title>Horarios de Apertura</title>
</head>
<body>
    <h1>Consulta de Horarios</h1>
    <form action="check_schedule.php" method="get">
        <label for="day">Day of the Week:</label>
        <select name="day" required>
            <option value="1">Monday</option>
            <option value="2">Tuesday</option>
            <option value="3">Wednesday</option>
            <option value="4">Thursday</option>
            <option value="5">Friday</option>
            <option value="6">Saturday</option>
            <option value="7">Sunday</option>
        </select>
        <br><br>
        <input type="submit" value="Check Schedule">
    </form>
</body>
</html>
```

Código PHP (check_schedule.php):

```php
<?php
// Recogemos el día seleccionado
$day = $_GET['day'];

// Mostramos el horario correspondiente
switch ($day) {
    case 1:
        echo "Monday: 9am - 5pm";
        break;
    case 2:
        echo "Tuesday: 9am - 5pm";
        break;
```

```
    case 3:
        echo "Wednesday: 9am - 5pm";
        break;
    case 4:
        echo "Thursday: 9am - 5pm";
        break;
    case 5:
        echo "Friday: 9am - 5pm";
        break;
    case 6:
        echo "Saturday: 10am - 4pm";
        break;
    case 7:
        echo "Sunday: Closed";
        break;
    default:
        echo "Invalid day selected.";
}
?>
```

Enunciado del Ejercicio 4: Horarios de Apertura de un Café Temático

Eres el desarrollador de la página web de un **café temático de videojuegos**. El café tiene diferentes **horarios de apertura** dependiendo del día de la semana. Tu tarea es desarrollar un sistema que permita a los usuarios seleccionar el día de la semana y les muestre el **horario correspondiente**.

Instrucciones:

1. Crea un formulario en HTML donde los usuarios puedan seleccionar un **día de la semana**.
2. Usa el método **GET** para enviar el día seleccionado al servidor.
3. En el servidor (archivo PHP), recoge el valor del día usando $_GET.
4. Usa un **switch** para mostrar el horario correspondiente:
    - Lunes a viernes: **9am - 8pm**.
    - Sábado: **10am - 6pm**.
    - Domingo: **Cerrado**.

Ejercicio 5: Galería de Imágenes Dinámica para un Blog de Viajes

Teoría: Bucle for en PHP

El bucle **for** se utiliza cuando necesitas ejecutar un bloque de código un número específico de veces. La estructura básica de un bucle for en PHP es:

```
for (inicialización; condición; incremento) {
    // Código a ejecutar en cada iteración
}
```

La **inicialización** se ejecuta una vez al principio, la **condición** se evalúa antes de cada iteración, y el **incremento** se ejecuta al final de cada iteración.

Ejemplo simple de for:

```
for ($i = 1; $i <= 5; $i++) {
    echo "Iteration " . $i . "<br>";
}
```

En este ejemplo, el bucle for imprime la palabra "Iteration" seguida del número de iteración, de 1 a 5.

Ejemplo Completo: Galería de Imágenes en un Blog

Supongamos que estás construyendo un **blog de viajes** y quieres mostrar una **galería de 5 imágenes**. Vamos a usar un bucle for para generar las etiquetas HTML necesarias para mostrar las imágenes.

Código HTML y PHP (gallery.php):

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <title>Galería de Viajes</title>
</head>
<body>
    <h1>Mi Galería de Viajes</h1>
    <?php
    for ($i = 1; $i <= 5; $i++) {
echo "<img src='images/travel".$i.".jpg' alt='Travel Image ".$i."'><br>";
    }
    ?>
</body>
</html>
```

Enunciado del Ejercicio 5: Galería de Arte Digital para un Evento de Diseño Gráfico

Imagina que estás creando una **galería de arte digital** para un evento de **diseño gráfico**. La galería debe mostrar una serie de imágenes que representan los trabajos de los participantes. Tu tarea es generar dinámicamente las imágenes usando un bucle for.

Instrucciones:

1. Crea una carpeta con 10 imágenes numeradas consecutivamente (1.jpg, 2.jpg, etc.).
2. Usa un **bucle for** para generar las etiquetas HTML que muestran las imágenes en la página.
3. Cada imagen debe tener una etiqueta **alt** que describa el trabajo del participante.

Ejercicio 6: Validación de Correos Electrónicos para una Competencia de Streaming

Teoría: El Uso del Bucle do...while en PHP

El bucle **do...while** es similar al bucle while, pero con una diferencia importante: **se ejecuta al menos una vez**, incluso si la condición es falsa desde el principio. La estructura básica es:

```
do {
    // Código que se ejecuta
} while (condición);
```

El código dentro del bloque do se ejecuta primero, y luego se evalúa la condición. Si la condición es verdadera, el bucle continúa ejecutándose; si es falsa, el bucle se detiene.

Ejemplo simple de do...while:

```php
<?php
$number = 0;

do {
    echo "Number: " . $number . "<br>";
    $number++;
} while ($number <= 5);
?>
```

En este ejemplo, el código se ejecuta al menos una vez, aunque la condición pueda ser falsa.

Ejemplo Completo: Validación de una Dirección de Correo Electrónico

Supongamos que estás desarrollando un sistema para registrar correos electrónicos de usuarios que se suscriben a un boletín. Necesitas asegurarte de que los correos ingresados sean válidos antes de aceptarlos. Usaremos un bucle do...while para seguir pidiendo al usuario que ingrese una dirección de correo hasta que sea válida.

Código HTML (formulario):

```html
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <title>Validación de Correo Electrónico</title>
</head>
<body>
```

```
    <h1>Registro de Correo Electrónico</h1>
    <form action="validate_email.php" method="get">
        <label for="email">Correo Electrónico:</label>
        <input type="email" name="email" required>
        <br><br>
        <input type="submit" value="Enviar">
    </form>
</body>
</html>
```

Código PHP (validate_email.php):

```
<?php
$email = $_GET['email'];
// Validación simple del correo electrónico
do {
    if (!filter_var($email, FILTER_VALIDATE_EMAIL)) {
        echo "Dirección de correo no válida. Inténtalo de nuevo.<br>";
        // Aquí podrías redirigir al usuario a la página del formulario
nuevamente o manejar la lógica de repetir
    } else {
        echo "Correo válido: " . $email;
    }
} while (!filter_var($email, FILTER_VALIDATE_EMAIL));
```

Enunciado del Ejercicio 6: Validación de Correos Electrónicos para Inscripción en un Torneo de Streaming

Imagina que estás creando un sistema de **inscripción en un torneo de streaming**, donde los jugadores deben proporcionar un **correo electrónico válido** para registrarse. El sistema debe asegurarse de que los correos sean válidos antes de aceptar la inscripción.

Instrucciones:

1. Crea un formulario en HTML donde los jugadores ingresen su **correo electrónico**.
2. Usa el método **GET** para enviar el correo electrónico al servidor.
3. En el servidor (archivo PHP), recoge el valor del correo electrónico usando $_GET.
4. Usa un **bucle do...while** para seguir pidiendo al usuario que ingrese un correo electrónico hasta que se introduzca uno válido. Puedes usar la función **filter_var()** para validar la dirección de correo.
5. Una vez que el correo electrónico sea válido, muestra un mensaje que diga **"Correo electrónico válido: [correo]"**.

Ejercicio 7: Encuesta de Opinión para una Convención de Tecnología

Teoría: El Bucle foreach en PHP

El bucle **foreach** es una estructura muy útil cuando necesitas recorrer todos los elementos de un array. A diferencia de for, no necesitas preocuparte por contar cuántos elementos tiene el array, ya que foreach recorrerá automáticamente todos los elementos.

La estructura básica de foreach es:

```
foreach ($array as $valor) {
    // Código que se ejecuta para cada valor
}
```

Si también necesitas la clave o índice del elemento, puedes utilizar:

```
foreach ($array as $clave => $valor) {
    // Código que se ejecuta para cada clave y valor
}
```

Ejemplo simple de foreach:

```
<?php
$fruits = ["Apple", "Banana", "Cherry"];

foreach ($fruits as $fruit) {
    echo "Fruit: " . $fruit . "<br>";
}
?>
```

En este ejemplo, el bucle foreach recorre el array de frutas y muestra cada uno de los valores.

Ejemplo Completo: Encuesta de Opinión en una Página Web

Supongamos que estás creando una página web que permite a los usuarios participar en una **encuesta de opinión**. Queremos mostrar varias opciones y permitir que los usuarios seleccionen una. Luego, con un bucle foreach, procesaremos las respuestas.

Código HTML (formulario):

```html
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <title>Encuesta de Opinión</title>
</head>
<body>
    <h1>Encuesta de Opinión</h1>
    <form action="process_poll.php" method="get">
        <label for="opinion">Selecciona una opción:</label>
        <select name="opinion" required>
            <option value="option1">Me encantó</option>
            <option value="option2">Estuvo bien</option>
            <option value="option3">Podría mejorar</option>
            <option value="option4">No me gustó</option>
        </select>
        <br><br>
        <input type="submit" value="Enviar">
    </form>
</body>
</html>
```

Código PHP (process_poll.php):

```php
<?php
// Recogemos la opinión seleccionada
$opinion = $_GET['opinion'];
// Creamos un array con las respuestas posibles
$responses = [
    "option1" => "Me encantó",
    "option2" => "Estuvo bien",
    "option3" => "Podría mejorar",
    "option4" => "No me gustó"
];
// Mostramos la respuesta seleccionada
foreach ($responses as $key => $value) {
    if ($key == $opinion) {
        echo "Has seleccionado: " . $value;
    }
}
?>
```

Enunciado del Ejercicio 7: Encuesta de Opinión para una Convención de Tecnología

Estás a cargo de organizar una **convención de tecnología** y deseas realizar una encuesta para recoger la opinión de los asistentes sobre los expositores. Cada asistente debe seleccionar una opción para evaluar el evento.

Instrucciones:

1. Crea un formulario en HTML donde los asistentes puedan seleccionar una **opinión sobre el evento**.
2. Usa el método **GET** para enviar la opinión seleccionada al servidor.
3. En el servidor (archivo PHP), recoge la opinión usando $_GET.
4. Usa un **bucle foreach** para recorrer las opciones de respuesta y mostrar la que el usuario ha seleccionado.

Ejercicio 8: Encuesta de Satisfacción para un Evento de eSports

Teoría: El Uso de la Declaración break en PHP

La declaración **break** se utiliza para salir de un bucle o de una estructura switch. Cuando el programa encuentra un break, interrumpe la ejecución del bucle o del switch y continúa con el código después de esa estructura.

La estructura básica de break en un bucle o switch es:

```
switch (variable) {
    case valor1:
        // Código
        break;
    case valor2:
        // Código
        break;
}
```

Ejemplo simple de switch con break:

```
$day = 3;

switch ($day) {
    case 1:
        echo "Today is Monday";
        break;
    case 2:
        echo "Today is Tuesday";
        break;
    case 3:
        echo "Today is Wednesday";
        break;
    default:
        echo "Invalid day";
}
```

Ejemplo Completo: Encuesta con foreach y break

Imagina que estás creando una encuesta donde los usuarios pueden evaluar el evento. Si seleccionan la opción **"Ninguna de las anteriores"**, el bucle se debe detener.

Código HTML (formulario):

```html
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <title>Encuesta de Satisfacción</title>
</head>
<body>
    <h1>Encuesta de Satisfacción</h1>
    <form action="process_survey.php" method="get">
        <label for="opinion">Selecciona una opción:</label>
        <select name="opinion" required>
            <option value="option1">Excelente</option>
            <option value="option2">Bueno</option>
            <option value="option3">Regular</option>
            <option value="option4">Ninguna de las anteriores</option>
        </select>
        <br><br>
        <input type="submit" value="Enviar">
    </form>
</body>
</html>
```

Código PHP (process_survey.php):

```php
<?php
// Recogemos la opinión seleccionada
$opinion = $_GET['opinion'];
// Creamos un array con las opciones de respuesta
$responses = [
    "option1" => "Excelente",
    "option2" => "Bueno",
    "option3" => "Regular",
    "option4" => "Ninguna de las anteriores"
];
// Recorremos el array de respuestas
foreach ($responses as $key => $value) {
    if ($key == $opinion) {
        if ($key == "option4") {
            echo "Gracias por tu tiempo, pero no has seleccionado ninguna
de las opciones.";
           break; // Detenemos el bucle si es "Ninguna de las anteriores"
        }
```

```
else {
            echo "Has seleccionado: " . $value;
        }
    }
}
?>
```

Enunciado del Ejercicio 8: Encuesta de Satisfacción para un Evento de eSports

Estás creando una **encuesta de satisfacción** para un evento de eSports. Si el usuario selecciona la opción "Ninguna de las anteriores", el bucle debe detenerse y se mostrará un mensaje especial.

Instrucciones:

1. Crea un formulario en HTML donde los usuarios puedan seleccionar una **opinión sobre el evento**.
2. Usa el método **GET** para enviar la opinión seleccionada al servidor.
3. En el servidor (archivo PHP), recoge la opinión usando $_GET.
4. Usa un **bucle foreach** para procesar las respuestas. Si el usuario selecciona "Ninguna de las anteriores", usa un **break** para detener el bucle y mostrar un mensaje de agradecimiento sin procesar las otras opciones.

Estos ejercicios introducen a los estudiantes a diferentes estructuras de control en PHP y los invitan a trabajar en situaciones más complejas y cercanas a la realidad que pueden encontrar en el desarrollo web profesional.

Ejercicio 9: Mostrar Eventos Próximos en una Convención de Tecnología

Teoría: El Uso de la Declaración continue en PHP

La declaración **continue** se utiliza para saltar el resto de las instrucciones en la iteración actual de un bucle y pasar a la siguiente iteración. A diferencia de break, que interrumpe por completo el bucle, continue solo omite el código que sigue en la iteración actual y continúa con la siguiente.

La estructura básica de continue en un bucle es:

```php
<?php

for ($i = 0; $i < 10; $i++) {
    if ($i % 2 == 0) {
        continue; // Si $i es un número par, saltamos a la siguiente
iteración
    }
    echo $i . "<br>";
}
```

En este ejemplo, el bucle se salta los números pares y solo imprime los números impares.

Ejemplo simple de foreach con continue:

```php
<?php
$items = ["Item 1", "Item 2", "Item 3", "Item 4", "Item 5"];

foreach ($items as $item) {
    if ($item == "Item 3") {
        continue; // Nos saltamos "Item 3"
    }
    echo $item . "<br>";
}
?>
```

Ejemplo Completo: Filtrar Eventos Pasados en una Página de Próximos Eventos

Supongamos que estás creando una página web que muestra **próximos eventos**. Queremos que solo se muestren los eventos futuros y que se omitan aquellos que ya hayan pasado. Usaremos la declaración continue para saltar la impresión de los eventos que ya han ocurrido.

Código HTML y PHP (mostrar_eventos.php):

```php
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <title>Próximos Eventos</title>
</head>
<body>
    <h1>Próximos Eventos</h1>

    <?php
    $eventos = [
        ["nombre" => "Evento 1", "fecha" => "2023-09-10"],
        ["nombre" => "Evento 2", "fecha" => "2023-12-01"],
        ["nombre" => "Evento 3", "fecha" => "2024-01-15"]
    ];

    $hoy = date("Y-m-d");

    foreach ($eventos as $evento) {
        if ($evento['fecha'] < $hoy) {
            continue; // Nos saltamos los eventos pasados
        }
        echo "Nombre del evento: " . $evento['nombre'] . "<br>";
        echo "Fecha del evento: " . $evento['fecha'] . "<br><br>";
    }
    ?>
</body>
</html>
```

En este código, cualquier evento cuya fecha sea anterior a la fecha actual ($hoy) será omitido gracias a la declaración continue.

Enunciado del Ejercicio 9: Filtrar Eventos de un Festival de Música

Imagina que eres el desarrollador de la página web para un **festival de música** que se celebra a lo largo del año. Necesitas crear una página que solo muestre los eventos futuros, omitiendo los eventos pasados.

Instrucciones:

1. Crea un archivo PHP que contenga un **array de eventos** con nombres y fechas.
2. Usa la función **date()** en PHP para obtener la fecha actual.
3. Usa un **bucle foreach** para recorrer los eventos.
4. Utiliza **continue** para saltar los eventos cuya fecha sea anterior a la fecha actual.
5. Muestra solo los eventos futuros con su nombre y fecha.

Ejercicio 10: Encuesta para Evaluar Conferencias en una Feria de Tecnología

Teoría: Manejo de Formularios y el Bucle foreach

En este ejercicio, vamos a crear una encuesta en la que los usuarios pueden seleccionar una conferencia que desean evaluar. La encuesta presentará diferentes opciones de conferencias y también incluirá la opción **"Ninguna de las anteriores"**. Para detener el procesamiento si se selecciona esta opción, usaremos una condición dentro del bucle para manejar la lógica.

Ejemplo Completo: Encuesta de Evaluación de Conferencias

Este ejemplo ilustra cómo evaluar las respuestas de los usuarios utilizando solo condiciones dentro del bucle foreach para gestionar las opciones.

Código HTML (formulario):

```html
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <title>Encuesta de Evaluación de Conferencias</title>
</head>
<body>
    <h1>Evaluación de Conferencias</h1>
    <form action="process_survey.php" method="get">
        <label for="conference">Selecciona la conferencia:</label>
        <select name="conference" required>
            <option value="conf1">Conferencia 1</option>
            <option value="conf2">Conferencia 2</option>
            <option value="none">Ninguna de las anteriores</option>
        </select>
        <br><br>
        <input type="submit" value="Enviar Evaluación">
    </form>
</body>
</html>
```

Código PHP (process_survey.php):

```php
<?php
// Recogemos la opción seleccionada del formulario
$conference = $_GET['conference'];

// Creamos un array con las opciones de conferencia
$conferences = [
    "conf1" => "Conferencia 1",
    "conf2" => "Conferencia 2",
    "none" => "Ninguna de las anteriores"
];

// Recorremos las opciones de conferencia usando foreach
foreach ($conferences as $key => $value) {
    // Verificamos si el usuario seleccionó "Ninguna de las anteriores"
    if ($key == "none" && $conference == "none") {
        echo "Gracias por tu tiempo, pero no has seleccionado ninguna conferencia.";
    }

    // Si se selecciona cualquier otra opción válida
    elseif ($key == $conference) {
        echo "Has evaluado: " . $value;
    }
}
?>
```

Enunciado del Ejercicio 10: Encuesta de Evaluación de Conferencias en una Feria de Tecnología

Estás desarrollando una **encuesta de evaluación de conferencias** para una feria de tecnología. Los asistentes deben seleccionar la conferencia que desean evaluar. Si seleccionan "Ninguna de las anteriores", se debe mostrar un mensaje de agradecimiento sin procesar otras opciones.

Instrucciones:

1. Crea un formulario en HTML donde los usuarios puedan seleccionar la **conferencia** que desean evaluar.
2. Usa el método **GET** para enviar la selección al servidor.
3. En el servidor (archivo PHP), recoge la selección usando $_GET.
4. Usa un **bucle foreach** para procesar las conferencias.
5. Si el usuario selecciona "Ninguna de las anteriores", muestra un mensaje de agradecimiento y no proceses otras opciones.

Pasos Detallados:

1. **Formulario en HTML**:
    - Crea un formulario con un campo de selección que permita al usuario elegir una conferencia o "Ninguna de las anteriores".
2. **Recogida de datos con GET**:
    - Usa el método **GET** para enviar los datos al servidor y recogerlos con $_GET['conference'] en el archivo PHP.
3. **Bucle foreach**:
    - Para detener el bucle, incluye una condición dentro del bucle que verifique si el usuario seleccionó "Ninguna de las anteriores".
    - Si se selecciona "Ninguna de las anteriores", muestra un mensaje de agradecimiento y no continúes procesando otras opciones.
4. **Visualización del resultado**:
    - Si el usuario selecciona cualquier otra opción válida, muestra un mensaje indicando cuál conferencia ha sido evaluada.

Refuerzo Teoría

1. Formularios y Método GET

El método GET es utilizado en formularios HTML para enviar datos al servidor. Es ideal cuando los datos que se envían no son sensibles, ya que los valores se transmiten como parámetros en la URL. La sintaxis básica para acceder a los datos enviados por GET en PHP es utilizando la variable superglobal $_GET:

```php
$value = $_GET['field_name']
```

El método GET tiene las siguientes características:

- Los datos se envían a través de la URL, visibles al usuario.
- Ideal para búsquedas, filtros o formularios que no requieran confidencialidad.
- Los datos enviados tienen un límite de longitud.
- Fácil de usar, ya que permite compartir la URL con los datos.

---

2. Condicionales en PHP

Las declaraciones condicionales nos permiten ejecutar código solo si se cumplen ciertas condiciones. Los condicionales en PHP incluyen:

a) if

El condicional if ejecuta un bloque de código si la condición es verdadera:

```php
if ($condition) {
    // Código que se ejecuta si $condition es verdadera
}
```

b) if...else

El condicional if...else evalúa una condición, ejecutando un bloque de código si es verdadera y otro si es falsa:

```php
if ($condition) {
    // Código si $condition es verdadera
} else {
    // Código si $condition es falsa
}
```

c) if...elseif...else

Este condicional permite evaluar múltiples condiciones:

```php
if ($condition1) {
    // Código si $condition1 es verdadera
} elseif ($condition2) {
    // Código si $condition2 es verdadera
} else {
    // Código si ninguna condición es verdadera
}
```

d) switch

El condicional switch evalúa una variable y ejecuta el código correspondiente al caso que coincida con su valor:

```php
switch ($variable) {
    case 'valor1':
        // Código para 'valor1'
        break;
    case 'valor2':
        // Código para 'valor2'
        break;
    default:
        // Código si no coincide ningún valor
}
```

---

3. Bucles en PHP

Los bucles permiten ejecutar repetidamente un bloque de código, ya sea un número determinado de veces o hasta que se cumpla una condición.

a) for

El bucle for es utilizado cuando se sabe cuántas veces debe ejecutarse el código:

```php
for ($i = 0; $i < 10; $i++) {
    // Código que se ejecuta 10 veces
}
```

b) while

El bucle while ejecuta un bloque de código mientras la condición sea verdadera:

```
while ($condition) {
    // Código que se ejecuta mientras $condition sea verdadera
}
```

c) do...while

El bucle do...while es similar a while, pero garantiza que el bloque de código se ejecute al menos una vez, independientemente de si la condición es verdadera o falsa:

```
do {
    // Código que se ejecuta al menos una vez
} while ($condition);
```

d) foreach

El bucle foreach es ideal para recorrer arrays o listas de elementos:

```
foreach ($array as $element) {
    // Código que se ejecuta para cada elemento del array
}
```

---

4. Control de Flujo: break y continue

En PHP, podemos controlar la ejecución de los bucles utilizando las declaraciones break y continue:

a) break

La declaración break detiene por completo la ejecución de un bucle o switch y continúa con el código después del bucle:

```
for ($i = 0; $i < 10; $i++) {
    if ($i == 5) {
        break; // El bucle se detiene cuando $i es igual a 5
    }
}
```

b) continue

La declaración continue salta el resto del código en la iteración actual y pasa directamente a la siguiente:

```php
for ($i = 0; $i < 10; $i++) {
    if ($i % 2 == 0) {
        continue; // Nos saltamos los números pares
    }
    echo $i . "<br>";
}
```