Tema 6: Acceso a APIS

Introducción

Las APIs (Interfaces de Programación de Aplicaciones) permiten que diferentes aplicaciones se comuniquen entre sí, compartan datos y funcionen de manera conjunta de manera eficiente. PHP, como un lenguaje de programación del lado del servidor ampliamente utilizado, ofrece herramientas y funciones integradas que facilitan la interacción con APIs.

Acceder a una API desde PHP implica enviar solicitudes HTTP, como GET o POST, a una URL específica, generalmente proporcionada por el proveedor de la API. Las respuestas, comúnmente en formato JSON o XML, se procesan en PHP para extraer la información necesaria.

Para facilitar este proceso, PHP cuenta con bibliotecas como cURL, que simplifican el manejo de solicitudes HTTP. Además, frameworks modernos como Laravel y Symfony ofrecen componentes dedicados para trabajar con APIs de manera más estructurada.

Cómo conectarnos a una API

Para realizar solicitudes a una API desde PHP, puedes utilizar la extensión cURL, que es ampliamente utilizada y está disponible en la mayoría de las instalaciones de PHP. Aquí hay un ejemplo básico de cómo realizar una solicitud GET a una API y manejar la respuesta:

```
<?php

// URL de la API que deseas consultar
$apiUrl = "https://api.ejemplo.com/data";

// Inicializar cURL
$curl = curl_init($apiUrl);

// Configurar opciones de cURL
curl_setopt($curl, CURLOPT_RETURNTRANSFER, true);

// Realizar la solicitud GET
$response = curl_exec($curl);

// Verificar si hubo errores en la solicitud
if(curl_errno($curl)){
    echo 'Error al realizar la solicitud: ' . curl_error($curl);
}

// Cerrar la sesión cURL
curl_close($curl);

// Manejar la respuesta JSON
$data = json_decode($response, true);

// Verificar si la decodificación fue exitosa
if ($data === null) {
    echo 'Error al decodificar la respuesta JSON.';
} else {
    // Trabajar con los datos devueltos
    var_dump($data);
}

?>
```

Este ejemplo utiliza cURL para realizar una solicitud GET a la URL de la API. La respuesta se almacena en la variable $response, y luego se decodifica con json_decode para trabajar con los datos en formato JSON.

Ten en cuenta que este es un ejemplo básico, y muchas APIs requieren autenticación. Para eso, puedes agregar opciones adicionales a cURL, como configurar encabezados con tokens de acceso o claves de API. Aquí hay un ejemplo de cómo agregar autenticación con un encabezado de clave de API:

```
// ... (código anterior)

// Configurar encabezados para la autenticación
$headers = [
    'Authorization: Bearer tu_clave_de_api',
    'Content-Type: application/json',
];

curl_setopt($curl, CURLOPT_HTTPHEADER, $headers);

// ... (resto del código)
```

Asegúrate de revisar la documentación específica de la API que estás utilizando para obtener detalles sobre cómo autenticarte y cómo se estructuran las respuestas. Además, ten en cuenta que algunos servicios pueden requerir otros métodos de solicitud, como POST, PUT o DELETE, que puedes configurar en cURL mediante curl_setopt.

Petición GET

Obtenemos los datos desde una URL determinada. Podemos usar la variable $data para imprimir la respuesta (echo $data) o retornar el valor de una función.

```
<?php
   $ch = curl_init();
   curl_setopt($ch, CURLOPT_URL, 'https://url-de-la-api');
   curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
   curl_setopt($ch, CURLOPT_HEADER, 0);
   $data = curl_exec($ch);
   curl_close($ch);
   //echo $data;
?>
```

Explicación línea por línea:

```
//Abrimos conexión cURL y la almacenamos en la variable $ch.
$ch = curl_init();

//Configuramos mediante CURLOPT_URL la URL de nuestra API
curl_setopt($ch, CURLOPT_URL, 'https://url-de-la-api');

//Abrimos conexión cURL y la almacenamos en la variable $ch.
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);

// 0 o 1, indicamos que no queremos al Header en nuestra respuesta
curl_setopt($ch, CURLOPT_HEADER, 0);

//Ejecuta la petición HTTP y almacena la respuesta en la variable $data.
$data = curl_exec($ch);
```

```
//Cerramos la conexión cURL
curl_close($ch);
```

Petición POST

Enviamos datos a una URL destino.

```
<?php
   $fields = array('field1' => 'valor1', 'field2' => urlencode('valor 2'));
   $fields_string = http_build_query($fields);
   $ch = curl_init();
   curl_setopt($ch, CURLOPT_URL, "https://url-de-la-web/test");
   curl_setopt($ch, CURLOPT_POST, 1);
   curl_setopt($ch, CURLOPT_POSTFIELDS, $fields_string );
   $data = curl_exec($ch);
   curl_close($ch);
?>
```

Explicación línea por línea:

```
//Inicializamos nuestro arreglo de valores
//Si nuestros valores contienen espacios o caracteres especiales
//podemos convertirlos mediante urlencode("valor");
$fields = array('field1' => 'valor1', 'field2' => urlencode('valor 2'));

//convertimos el arreglo en formato URL
$fields_string = http_build_query($fields);

//Abrimos la conexión cURL
$ch = curl_init();

//Configuramos mediante CURLOPT_URL la URL destino
curl_setopt($ch, CURLOPT_URL, "https://url-de-la-web/test");

//Indicamos que se trata de una petición POST con valor "1" o "true"
curl_setopt($ch, CURLOPT_POST, 1);

//Asignamos los campos a enviar en el POST
curl_setopt($ch, CURLOPT_POSTFIELDS, $fields_string );

//Ejecuta la petición HTTP y almacena la respuesta en la variable $data.
$data = curl_exec($ch);

//Cierra la conexión cURL
curl_close($ch);
```

Ejemplo completo de llamada a API y obtención de datos

Vamos a hacer una llamada a una API que nos da datos sobre el COVID en Méjico:

```
<?php

$curl = curl_init(); //inicia la sesión cURL

curl_setopt_array($curl, array(
        CURLOPT_URL => "https://covid-19-coronavirus-statistics.p.rapidapi.com/v1/stats?
country=Mexico", //url a la que se conecta
```

```
        CURLOPT_RETURNTRANSFER => true, //devuelve el resultado como una cadena del tipo curl_exec

        CURLOPT_FOLLOWLOCATION => true, //sigue el encabezado que le envíe el servidor

        CURLOPT_ENCODING => "", // permite decodificar la respuesta y puede ser"identity", "deflate", y "gzip", si está vacío recibe todos los disponibles.

        CURLOPT_MAXREDIRS => 10, // Si usamos CURLOPT_FOLLOWLOCATION le dice el máximo de encabezados a seguir

        CURLOPT_TIMEOUT => 30, // Tiempo máximo para ejecutar

        CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1, // usa la versión declarada

        CURLOPT_CUSTOMREQUEST => "GET", // el tipo de petición, puede ser PUT, POST, GET o Delete dependiendo del servicio

        CURLOPT_HTTPHEADER => array(
                "x-rapidapi-host: covid-19-coronavirus-statistics.p.rapidapi.com",
                "x-rapidapi-key: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"
        ), //configura las cabeceras enviadas al servicio
)); //curl_setopt_array configura las opciones para una transferencia cURL

$response = curl_exec($curl);// respuesta generada

$err = curl_error($curl); // muestra errores en caso de existir

curl_close($curl); // termina la sesión

if ($err) {
        echo "cURL Error #:" . $err; // mostramos el error
} else {
        echo $response; // en caso de funcionar correctamente
}
```

cURL es una librería que permite realizar peticiones HTTP con el objetivo de transferir información, permite armar un script que literalmente se comporte como un navegador para así realizar una petición a otro servidor remoto. Pero puede ocurrir que no nos funcione en nuestro equipo. Para ello hacemos una pequeña comprobación.

Antes de empezar debemos tener activado cURL en nuestro servidor si lo tenemos en local asegurarnos que nuestro php.ini no tenga comentada la opción (quitando las comillas que se muestran en el siguiente código) y esté instalado:

```
;extension=php_curl.dll
```

Ya que en caso contrario nos mostrará un error parecido al siguiente:

```
<b>Fatal error</b>:  Uncaught Error: Call to undefined function curl_init()
```

Suponiendo que todo esté correcto nos regresará el siguiente json:

{"error":false,"statusCode":200,"message":"OK","data":{"lastChecked":"2020-04-11T20:06:20.272Z","covid19Stats":[{"city":"","province":"","country":"Mexico","lastUpdate":"2020-04-10 22:53:48","keyId":"Mexico","confirmed":3441,"deaths":194,"recovered":633}]}}

Con ello sabemos que nuestro archivo funciona, de acuerdo a la documentación los parámetros son opcionales por lo que podemos eliminar ?country=Mexico y nos dará una lista completa en json.

Si bien ya tenemos una respuesta o varias dependiendo de los parámetros que agreguemos no es tan funcional como para agregarlo así que vamos a parsear el código para tener un resultado más agradable.

Parseando JSON con PHP

PHP nos permite pasear json fácilmente o crear uno nuevo utilizando json_decode() al utilizar la función podemos obtener un objeto o un arreglo simple. Si aplicamos json_decode() sin parámetros lo tomará como false, y nos regresa un objeto como el siguiente:

```
json_decode($response);
stdClass Object
(
   [error] =>
   [statusCode] => 200
   [message] => OK
   [data] => stdClass Object
      (
         [lastChecked] => 2020-04-11T21:06:19.985Z
         [covid19Stats] => Array
            (
               [0] => stdClass Object
                  (
                     [city] =>
                     [province] =>
                     [country] => Mexico
                     [lastUpdate] => 2020-04-10 22:53:48
                     [keyId] => Mexico
                     [confirmed] => 3441
                     [deaths] => 194
                     [recovered] => 633
                  )

            )

      )

)
```

Ya con nuestro objeto podemos recuperar los datos que queremos por ejemplo fecha de actualización (lastUpdate).

```
$objeto = json_decode($response);
$objeto->data->covid19Stats[0]->lastUpdate;
```

Si queremos podemos sacar los datos y acomodarlos de acuerdo a nuestro diseño o necesidad, incluso almacenarlos en una base de datos para usar después, en este ejemplo solamente presentaremos los datos de esta forma:

```
COVID19 en Mexico
Actualizado: 2020-04-10 22:53:48
Confirmados: 3441
Muertos: 194
Recuperados: 633
```

Les comparto el código completo sin clave API para que generen la suya y hagan las modificaciones que necesiten:

```
<?php

$curl = curl_init();

curl_setopt_array($curl, array(
	CURLOPT_URL => "https://covid-19-coronavirus-statistics.p.rapidapi.com/v1/stats?country=Mexico",
	CURLOPT_RETURNTRANSFER => true,
	CURLOPT_FOLLOWLOCATION => true,
	CURLOPT_ENCODING => "",
	CURLOPT_MAXREDIRS => 10,
	CURLOPT_TIMEOUT => 30,
	CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1,
	CURLOPT_CUSTOMREQUEST => "GET",
	CURLOPT_HTTPHEADER => array(
		"x-rapidapi-host: covid-19-coronavirus-statistics.p.rapidapi.com",
		"x-rapidapi-key: APIKEYAQUÍ"
	),
));

$response = curl_exec($curl);
$err = curl_error($curl);

curl_close($curl);

if ($err) {
	echo "cURL Error #:" . $err;
} else {
	$objeto = json_decode($response);

	echo "COVID19 en ".$objeto->data->covid19Stats[0]->country."<br/>";
	echo "Actualizado: ".$objeto->data->covid19Stats[0]->lastUpdate."<br/>";
	echo "Confirmados: ".$objeto->data->covid19Stats[0]->confirmed."<br/>";
	echo "Muertos: ".$objeto->data->covid19Stats[0]->deaths."<br/>";
	echo "Recuperados: ".$objeto->data->covid19Stats[0]->recovered."<br/>";

}
```

Webs de APIs de testeo

Podemos practicar con un buen número de APIs gratuitas de las que recibir contenidos. Un par de webs donde podemos encontrar estas APIs son

https://apipheny.io/free-api/

https://rapidapi.com/collection/list-of-free-apis