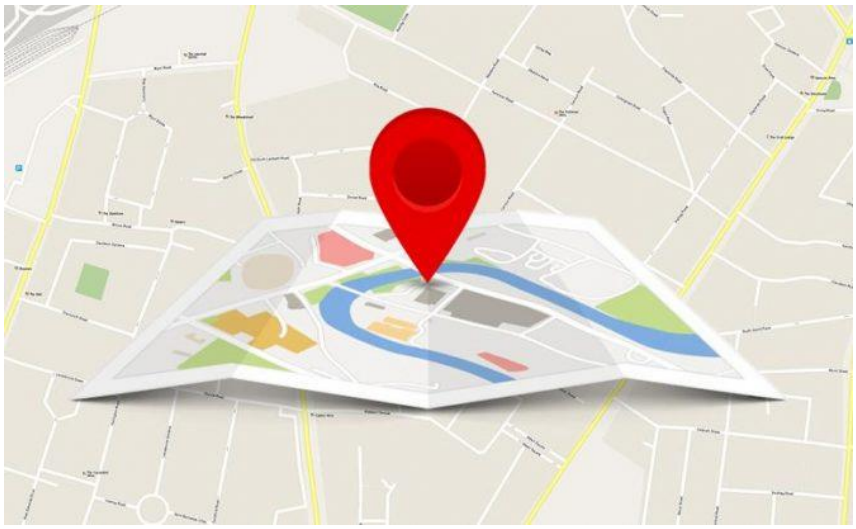
Localizar direcciones en Google Maps con PHP



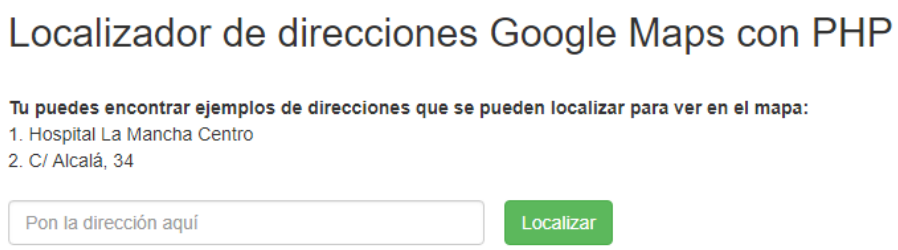
Hoy en día es muy habitual que cualquier página web cuente con un mapa en el que se muestra la dirección de una tienda, unas oficinas o la posibilidad de indicar una dirección para que se muestre en el mapa. Todo este proceso se conoce con el nombre de geocodificación, es decir, convertir una dirección en coordenadas geográficas con el fin de mostrarla dentro de un mapa. Para conseguir esto, la API de Google Maps puede sernos de gran ayuda. Lo único que debemos hacer es pasar esa dirección a la API para que esta nos devuelva la respuesta en formato JSON o XML. A lo largo de nuestro White Paper veremos todo este proceso y lo fácil que resulta hacerlo mediante PHP.

Para llevar a cabo esto, lo primero que deberéis hacer es conseguir es una Key API de Google para empezar a utilizarla. Esto se puede conseguir en la siguiente web: https://cloud.google.com/maps-platform/. Ahí se deberán seguir los pasos que se indican para su creación.

Veamos a continuación el código que debemos utilizar para poder mostrar una dirección en un mapa.

Archivo index.php

Este archivo será el encargado de pintar el formulario donde el usuario podrá colocar la dirección que quiere que aparezca en el mapa. Tendrá un aspecto parecido al que se muestra en la siguiente imagen.



El código encargado de pintar ese formulario sería el siguiente.

HTML

```
<div>

  <div><strong>Tú puedes encontrar ejemplos de direcciones que se pueden localizar para ver en el mapa:</strong></div>

  <div>1. Hospital La Mancha Centro</div>

  <div>2. C/ Alcalá, 34</div>

</div>

<br>

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

```
    <div class="row">
      <div class="col-sm-4">
        <div class="form-group">
          <input type='text' name='searchAddress' class="form-control" placeholder='Pon la dirección aquí'
/>
        </div>
      </div>
      <div class="form-group">
        <input type='submit' value='Localizar' class="btn btn-success" />
      </div>
    </div>
 </form>
```

Este código está dividido en dos bloques. El primero se encarga de pintar lo que es la información que aparece encima del formulario, mientras que el segundo bloque muestra el formulario donde hay que escribir la dirección. Este formulario está formado por una caja de texto y un botón que se encarga de enviar la petición. Como se puede observar, se trata de un formulario muy sencillo y que no tiene ningún misterio. Dentro de este archivo index.php también colocaremos el código que se encargará de pintar la dirección en el mapa. El código sería el siguiente.

PHP y JAVASCRIPT

```
<?php
   if($_POST) {
     $geocodeData = getGeocodeData($_POST['searchAddress']);
     if($geocodeData) {
       $latitude = $geocodeData[0];
       $longitude = $geocodeData[1];
       $address = $geocodeData[2];
?>
<div id="gmap">Cargando mapa...</div>
<script type="text/javascript" src="https://maps.google.com/maps/api/js?key=API-KEY"></script>
```

```
<script type="text/javascript">
  function init_map() {
    var options = {
      zoom: 14,
      center: new google.maps.LatLng(<?php echo $latitude; ?>, <?php echo $longitude; ?>),
      mapTypeId: google.maps.MapTypeId.ROADMAP
    };
    map = new google.maps.Map($("#gmap")[0], options);
    marker = new google.maps.Marker({
      map: map,
      position: new google.maps.LatLng(<?php echo $latitude; ?>, <?php echo $longitude; ?>)
    });
    infowindow = new google.maps.InfoWindow({
      content: "<?php echo $address; ?>"
    });
    google.maps.event.addListener(marker, "click", function () {
      infowindow.open(map, marker);
    });
  }
  google.maps.event.addDomListener(window, 'load', init_map);
</script>
<?php
    } else {
      echo "Detalles incorrectos!";
    }
}
?>
```

En este código, lo primero que hacemos es comprobar si se ha pulsado el botón de envío del formulario. Esto lo hacemos verificando si existe la variable de PHP "$_POST". En esta variable es donde estará toda la información enviada desde el formulario.

PHP

```php
if($_POST) {
```

Si se ha pulsado el botón, lo siguiente que hacemos es llamar a la función "getGeocodeData", que explicaremos posteriormente, y al que se le pasa como parámetro la dirección escrita en el formulario.

PHP

```php
$geocodeData = getGeocodeData($_POST['searchAddress']);
```

Esta función devuelve la información transmitida por Google y se la asigna a una variable. Si esta variable contiene información, entonces lo que hacemos es sacar la latitud, la longitud y la dirección que se mostrará en el mapa.

PHP

```php
if($geocodeData) {
  $latitude = $geocodeData[0];
  $longitude = $geocodeData[1];
  $address = $geocodeData[2];
```

En el caso de que no devuelva nada, se mostrará un mensaje de error. Esto se muestra en el código anterior al final del todo, con el "else".

PHP

```php
    } else {
     echo "Detalles incorrectos!";
  }
```

Lo siguiente que aparece en el código es el contenedor donde se pintará el mapa con la información devuelta por Google.

HTML

```html
<div id="gmap">Cargando mapa...</div>
```

También incluimos la llamada al API de Google para poder hacer uso de sus mapas.

HTML

```
<script type="text/javascript" src="https://maps.google.com/maps/api/js?key=API-KEY"></script>
```

En la línea anterior deberíais cambiar “API-KEY” por la cadena de caracteres que tenéis que crear en la página de Google que os hemos indicado más arriba.

Lo siguiente ya es hacer uso del código que se encarga de pintar el mapa y que se trata de funciones que ofrece Google para trabajar con sus mapas.

Aquí lo primero que hacemos es crear un array de variables de configuración del mapa que se mostrará.

JAVASCRIPT

```
var options = {
   zoom: 14,
   center: new google.maps.LatLng(<?php echo $latitude; ?>, <?php echo $longitude; ?>),
   mapTypeId: google.maps.MapTypeId.ROADMAP
};
```

Las opciones que aquí configuramos son:

- **zoom**: Se indica el valor del zoom que queremos que tenga el mapa. Cuanto más bajo sea el valor, menor será el zoom.
- **center**: Aquí se indica la latitud y la longitud de la dirección que queremos que se muestre en el mapa.
- **mapTypeId**: El tipo de mapa que queremos mostrar.

La siguiente línea se encarga de crear el objeto del mapa y asignarlo al contenedor que hemos creado antes dentro de nuestra página web.

JAVASCRIPT

```
map = new google.maps.Map($("#gmap")[0], options);
```

Creamos un marcador que señale el punto que hemos solicitado que se muestre en el mapa.

JAVASCRIPT

```
marker = new google.maps.Marker({
```

```
    map: map,
    position: new google.maps.LatLng(<?php echo $latitude; ?>, <?php echo $longitude; ?>)
});
```

También creamos una ventana de información que se mostrará cuando se pulse sobre el marcador. En esta se podrá ver la dirección.

JAVASCRIPT

```
infowindow = new google.maps.InfoWindow({
    content: "<?php echo $address; ?>"
});
```

Para que esta ventana informativa se abra, es necesario crear un evento que al pulsar sobre la marca se visualice el mensaje.

JAVASCRIPT

```
google.maps.event.addListener(marker, "click", function () {
    infowindow.open(map, marker);
});
```

Archivo functions.php

Dentro de este archivo estará la función “getGeocodeData” que hemos utilizado en más arriba. Como hemos dicho, se encarga de hacer la llamada a Google con los datos pasados en el formulario para que nos devuelva la latitud, longitud y demás información asociada a esa petición. Nosotros la hemos colocado en un archivo externo, pero perfectamente podría estar dentro del index.php.

El código de este método es el siguiente.

PHP

```
function getGeocodeData($address) {
  $address = urlencode($address);
```

```
    $googleMapUrl = "https://maps.googleapis.com/maps/api/geocode/json?address={$address}&key=API-KEY";

    $geocodeResponseData = file_get_contents($googleMapUrl);

    $responseData = json_decode($geocodeResponseData, true);

    if($responseData['status']=='OK') {

        $latitude = isset($responseData['results'][0]['geometry']['location']['lat']) ? $responseData['results'][0]['geometry']['location']['lat'] : "";

        $longitude = isset($responseData['results'][0]['geometry']['location']['lng']) ? $responseData['results'][0]['geometry']['location']['lng'] : "";

        $formattedAddress = isset($responseData['results'][0]['formatted_address']) ? $responseData['results'][0]['formatted_address'] : "";

        if($latitude && $longitude && $formattedAddress) {

            $geocodeData = array();

            array_push(

                $geocodeData,

                $latitude,

                $longitude,

                $formattedAddress

            );

            return $geocodeData;

        } else {

            return false;

        }

    } else {

        echo "ERROR: {$responseData['status']}";

        return false;

    }

}
```

Lo primero que hacemos en este método es hacer uso de la función “urlencode” de PHP. Esta función es recomendable utilizarla cuando se codifica una cadena a ser usada como la parte de consulta de una URL como va a ser nuestro caso.

PHP

```
$address = urlencode($address);
```

Después generamos la url con la llamada al API de Google y que nos devolverá la información de la dirección indicada en el formulario.

PHP

```
$googleMapUrl = "https://maps.googleapis.com/maps/api/geocode/json?address={$address}&key=API-KEY";
```

Aquí debemos sustituir “API-KEY” por nuestra clave de Google, igual que hemos comentado anteriormente.

La siguiente línea de código se encargará de recuperar la información ofrecida por Google

PHP

```
$geocodeResponseData = file_get_contents($googleMapUrl);
```

Mediante la función “json_decode” de PHP convertimos la respuesta devuelta por el API de Google en un array de PHP con el que poder trabajar.

PHP

```
$responseData = json_decode($geocodeResponseData, true);
```

En el caso de que el valor del campo “STATUS” sea “OK”, podremos recuperar la información de la consulta realizada. Los valores de la latitud, longitud y dirección, serán colocados en un array que devolverá el método. En el caso de que el valor “STATUS” no sea “OK”, el método devolverá el código asignado a ese campo, impidiendo de esta forma que se pinte el mapa en nuestra página web.

PHP

```
if($responseData['status']=='OK') {

    $latitude = isset($responseData['results'][0]['geometry']['location']['lat']) ? $responseData['results'][0]['geometry']['location']['lat'] : "";

    $longitude = isset($responseData['results'][0]['geometry']['location']['lng']) ? $responseData['results'][0]['geometry']['location']['lng'] : "";
```

```
    $formattedAddress = isset($responseData['results'][0]['formatted_address']) ? $responseData['results'][0]['formatted_address'] : "";

    if($latitude && $longitude && $formattedAddress) {

      $geocodeData = array();

      array_push(

        $geocodeData,

        $latitude,

        $longitude,

        $formattedAddress

      );

      return $geocodeData;

    } else {

      return false;

    }

  } else {

    echo "ERROR: {$responseData['status']}";

    return false;

  }
```

Este código visto a lo largo de nuestro White Paper os puede servir como referencia a la hora de implementar esta funcionalidad en cualquier desarrollo web.