

Para crear un script en PHP que use la API de Gmail para leer correos y extraer adjuntos de correos específicos, debes seguir los siguientes pasos:

1. Crear un proyecto en la Consola de Desarrolladores de Google y habilitar la API de Gmail para ese proyecto. Esto te dará acceso a las credenciales necesarias para acceder a la API.
2. Descargar la biblioteca de PHP de Google API para acceder a las funciones de la API de Gmail desde PHP.
 - a. La biblioteca de PHP de Google API se puede descargar desde el repositorio oficial en GitHub:
<https://github.com/googleapis/google-api-php-client>
 - b. También puedes instalar la biblioteca utilizando Composer, un administrador de paquetes para PHP. Para hacerlo, debes agregar el siguiente comando en tu archivo composer.json:

```
"require": {  
  
    "google/apiclient": "^2.0"  
  
}
```

Luego, ejecutas el comando `composer install` en la línea de comandos para instalar la biblioteca y sus dependencias en tu proyecto.

Después de descargar la biblioteca, debes incluir el archivo "autoload.php" para cargar automáticamente las clases necesarias cuando sea necesario en tu proyecto de PHP.

3. Autenticar la aplicación de PHP con la cuenta de Google que tiene acceso al correo que quieres leer. Para evitar tener que autorizar la aplicación cada vez que se ejecuta el script, debes usar las credenciales del cliente para obtener un token de acceso que se almacena en un archivo en la máquina donde se ejecuta el script.
 - a. Para autenticar la aplicación de PHP con la cuenta de Google que tiene acceso al correo que quieres leer, debes seguir los siguientes pasos:
 - i. Crea un proyecto en la Consola de Desarrolladores de Google y habilita la API de Gmail para ese proyecto. Esto te dará acceso a las credenciales necesarias para acceder a la API.
 - ii. Crea un archivo JSON con las credenciales de cliente de la API de Google. Para hacer esto, debes seguir los siguientes pasos:
 - a. En la Consola de Desarrolladores de Google, abre la

sección "Credenciales" de tu proyecto.

b. Haz clic en el botón "Crear credenciales" y selecciona "ID de cliente de OAuth".

c. Selecciona "Aplicación web" como el tipo de aplicación y establece un nombre para la credencial.

d. Agrega las URIs de redireccionamiento de tu aplicación web. En este caso, debes agregar la URL donde se ejecutará tu script de PHP.

e. Haz clic en "Crear" para crear la credencial.

f. Descarga las credenciales en formato JSON haciendo clic en el botón "Descargar" en la fila de la credencial correspondiente.

- iii. En tu script de PHP, crea un objeto de cliente de Google API con las credenciales de cliente descargadas. Para hacer esto, debes seguir los siguientes pasos:

```
// Incluye la biblioteca de Google API para PHP
```

```
require_once
```

```
'ruta/a/google-api-php-client/vendor/autoload.php';
```

```
// Carga las credenciales de cliente desde un archivo JSON
```

```
$client = new Google_Client();
```

```
$client->setAuthConfig('ruta/a/credenciales.json');
```

- iv. Usa el objeto de cliente de Google API para obtener un token de acceso que se almacenará en un archivo en la máquina donde se ejecuta el script. Este token de acceso permitirá que tu script acceda a los recursos de la cuenta de Google sin tener que pedir al usuario que autorice la aplicación cada vez que se ejecute el script. Para obtener el token de acceso, debes seguir los siguientes pasos:

```
// Establece el alcance de la API de Gmail que se va a usar
```

```
$client->setScopes(Google_Service_Gmail::GMAIL_READONLY);

// Define la ruta donde se almacenará el token de acceso

$tokenPath = 'ruta/a/token.json';

// Si el token de acceso ya existe, carga el token desde el
archivo

if (file_exists($tokenPath)) {

    $accessToken = json_decode(file_get_contents($tokenPath),
true);

    $client->setAccessToken($accessToken);
}

// Si no hay token, solicita la autorización del usuario

if ($client->isAccessTokenExpired()) {

    $authUrl = $client->createAuthUrl();

    printf("Abra la siguiente URL e ingrese el código de
autorización:\n%s\n", $authUrl);

    $authCode = trim(fgets(STDIN));

    $accessToken =
$client->fetchAccessTokenWithAuthCode($authCode);

    file_put_contents($tokenPath, json_encode($accessToken));

    $client->setAccessToken($accessToken);
}
```

- v. En resumen, estos son los pasos para autenticar una aplicación de PHP con la API de Gmail de Google. Una vez que la aplicación esté autenticada, podrás acceder a los correos electrónicos y realizar cualquier otra acción permitida por la API de Gmail.
- b. Para usar las credenciales del cliente y obtener un token de acceso que se almacena en un archivo en la máquina donde se ejecuta el script, debes seguir los siguientes pasos:
 - i. Carga las credenciales de cliente desde un archivo JSON. En este ejemplo, el archivo JSON se llama "credenciales.json" y se encuentra en la misma carpeta que el script de PHP:

```
// Incluye la biblioteca de Google API para PHP

require_once
'ruta/a/google-api-php-client/vendor/autoload.php';

// Carga las credenciales de cliente desde un archivo JSON

$client = new Google_Client();

$client->setAuthConfig('credenciales.json');
```

- ii. Establece el alcance de la API de Gmail que se va a usar. En este ejemplo, se utiliza el alcance de solo lectura de Gmail:

```
$client->setScopes(Google_Service_Gmail::GMAIL_READONLY);
```

- iii. Define la ruta donde se almacenará el token de acceso. En este ejemplo, el archivo que contendrá el token se llama "token.json" y se encuentra en la misma carpeta que el script de PHP:

```
$tokenPath = 'token.json';
```

- iv. Si el token de acceso ya existe, cárgalo desde el archivo:

```
if (file_exists($tokenPath)) {
```

```
$accessToken = json_decode(file_get_contents($tokenPath),  
true);  
  
$client->setAccessToken($accessToken);  
  
}
```

- v. Si no hay token, solicita la autorización del usuario y obtén el token de acceso:

```
if ($client->isAccessTokenExpired()) {  
  
    // Solicita la autorización del usuario  
  
    $authUrl = $client->createAuthUrl();  
  
    printf("Abra la siguiente URL e ingrese el código de  
autorización:\n%s\n", $authUrl);  
  
    $authCode = trim(fgets(STDIN));  
  
  
    // Intercambia el código de autorización por un token de  
acceso  
  
    $accessToken =  
$client->fetchAccessTokenWithAuthCode($authCode);  
  
  
  
    // Almacena el token de acceso en un archivo  
  
    file_put_contents($tokenPath, json_encode($accessToken));  
  
  
    // Asigna el token de acceso al cliente  
  
    $client->setAccessToken($accessToken);  
  
}
```

- c. Una vez que hayas seguido estos pasos, el token de acceso se almacenará en el archivo especificado y se utilizará para autenticar la aplicación de PHP con la cuenta de Google
- d. Para obtener las credenciales de cliente desde un archivo JSON y realizar la carga de dichas credenciales, debes seguir los siguientes pasos:
 - i. Crea un proyecto en la consola de desarrolladores de Google y habilita la API de Gmail. Esto te permitirá obtener las credenciales de cliente necesarias para autenticar tu aplicación de PHP.
 - ii. Descarga las credenciales de cliente en formato JSON desde la consola de desarrolladores de Google. Asegúrate de que el archivo de credenciales JSON esté en una ubicación segura y accesible en el servidor donde se ejecutará tu script de PHP.
 - iii. En tu script de PHP, carga las credenciales de cliente desde el archivo JSON utilizando el método `setAuthConfig` del objeto `Google_Client`. El siguiente ejemplo muestra cómo cargar las credenciales desde un archivo llamado `credentials.json` ubicado en la misma carpeta que el script de PHP:

```
// Incluye la biblioteca de Google API para PHP
```

```
require_once  
'ruta/a/google-api-php-client/vendor/autoload.php';
```

```
// Carga las credenciales de cliente desde un archivo JSON
```

```
$client = new Google_Client();  
  
$client->setAuthConfig('credentials.json');
```

- iv. Verifica que las credenciales se hayan cargado correctamente. Puedes hacer esto imprimiendo los datos del objeto `Google_Client` utilizando el método `getAuthConfig()`:

```
var_dump($client->getAuthConfig());
```

- v. Si las credenciales se cargaron correctamente, deberías ver una salida similar a la siguiente:

```
array(5) {
```

```
[ "web"]=>

array(6) {

    ["client_id"]=>

        string(64)
        "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx.apps.g
oogleusercontent.com"

    ["project_id"]=>

        string(32) "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx"

    ["auth_uri"]=>

        string(50) "https://accounts.google.com/o/oauth2/auth"

    ["token_uri"]=>

        string(55) "https://oauth2.googleapis.com/token"

    ["auth_provider_x509_cert_url"]=>

        string(43) "https://www.googleapis.com/oauth2/v1/certs"

    ["client_secret"]=>

        string(64) "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx"

}

...

}
```

- vi. Una vez que hayas seguido estos pasos, habrás obtenido las credenciales de cliente desde un archivo JSON y las habrás cargado en tu script de PHP.
4. Utilizar la API de Gmail de Google para obtener una lista de los correos electrónicos en la cuenta que cumplan con ciertos criterios, por ejemplo, correos con un asunto específico o enviados por un remitente específico.

5. Analizar los correos electrónicos seleccionados para extraer los archivos adjuntos.
6. Guardar los archivos adjuntos en una carpeta específica en la máquina donde se ejecuta el script.
7. Automatizar el proceso utilizando una tarea programada en el sistema operativo de la máquina donde se ejecuta el script.

El siguiente es un ejemplo básico de cómo se podría escribir un script de PHP para extraer archivos adjuntos de correos electrónicos específicos en Gmail:

```
<?php
// Incluir la biblioteca de Google API para PHP
require_once 'ruta/a/google-api-php-client/vendor/autoload.php';

// Cargar las credenciales de cliente desde un archivo JSON
$client = new Google_Client();
$client->setAuthConfig('ruta/a/credenciales.json');
$client->setAccessType('offline');

// Autenticar el cliente con las credenciales del usuario
$tokenPath = 'ruta/a/token.json';
if (file_exists($tokenPath)) {
    $accessToken = json_decode(file_get_contents($tokenPath), true);
    $client->setAccessToken($accessToken);
}

// Si no hay token, solicitar la autorización del usuario
if ($client->isAccessTokenExpired()) {
    $authUrl = $client->createAuthUrl();
    printf("Abra la siguiente URL e ingrese el código de autorización:\n%s\n", $authUrl);
    $authCode = trim(fgets(STDIN));
    $accessToken = $client->fetchAccessTokenWithAuthCode($authCode);
    file_put_contents($tokenPath, json_encode($accessToken));
    $client->setAccessToken($accessToken);
}

// Crear un objeto de Gmail API y obtener los mensajes que cumplen con ciertos
criterios
$service = new Google_Service_Gmail($client);
$optParams = array(
    'q' => 'subject: "asunto del correo" from: "remitente@dominio.com"',
    'maxResults' => 10
```



```
);  
$results = $service->users_messages->listUsersMessages('me', $optParams);  
  
// Recorrer los mensajes y extraer los archivos adjuntos  
foreach ($results->getMessages() as $message) {  
    $msg = $service->users_messages->get('me', $message->getId());  
    $payload = $msg->getPayload();  
    foreach ($payload->getParts() as $part) {  
        if ($part->getFilename() && $part->getBody()) {  
            $data = $part->getBody()->getData();  
            $fileData = base64_decode($data);  
            $fileName = $part->getFilename();  
            // Guardar el archivo en una carpeta  
            file_put_contents("ruta/a/carpeta/{$fileName}", $fileData);  
        }  
    }  
}  
  
?>
```