<?php  
  
/\*\*  
 \* This sample uploads and sets a custom thumbnail for a video.  
 \*  
 \* 1. It uploads an image using the "Google\_MediaFileUpload" class.  
 \* 2. It sets the uploaded image as a custom thumbnail to the video by  
 \*    calling the API's "youtube.thumbnails.set" method  
 \*  
 \* @author Ibrahim Ulukaya  
\*/  
  
/\*\*  
 \* Library Requirements  
 \*  
 \* 1. Install composer (https://getcomposer.org)  
 \* 2. On the command line, change to this directory (api-samples/php)  
 \* 3. Require the google/apiclient library  
 \*    $ composer require google/apiclient:~2.0  
 \*/  
if (!file\_exists(\_\_DIR\_\_ . '/vendor/autoload.php')) {  
  throw new \Exception('please run "composer require google/apiclient:~2.0" in "' . \_\_DIR\_\_ .'"');  
}  
  
require\_once \_\_DIR\_\_ . '/vendor/autoload.php';  
session\_start();  
  
/\*  
 \* You can acquire an OAuth 2.0 client ID and client secret from the  
 \* Google API Console <https://console.cloud.google.com/>  
 \* For more information about using OAuth 2.0 to access Google APIs, please see:  
 \* <https://developers.google.com/youtube/v3/guides/authentication>  
 \* Please ensure that you have enabled the YouTube Data API for your project.  
 \*/  
$OAUTH2\_CLIENT\_ID = 'REPLACE\_ME';  
$OAUTH2\_CLIENT\_SECRET = 'REPLACE\_ME';  
  
$client = new Google\_Client();  
$client->setClientId($OAUTH2\_CLIENT\_ID);  
$client->setClientSecret($OAUTH2\_CLIENT\_SECRET);  
$client->setScopes('https://www.googleapis.com/auth/youtube');  
$redirect = filter\_var('http://' . $\_SERVER['HTTP\_HOST'] . $\_SERVER['PHP\_SELF'],  
    FILTER\_SANITIZE\_URL);  
$client->setRedirectUri($redirect);  
  
// Define an object that will be used to make all API requests.  
$youtube = new Google\_Service\_YouTube($client);  
  
// Check if an auth token exists for the required scopes  
$tokenSessionKey = 'token-' . $client->prepareScopes();  
if (isset($\_GET['code'])) {  
  if (strval($\_SESSION['state']) !== strval($\_GET['state'])) {  
    die('The session state did not match.');  
  }  
  
  $client->authenticate($\_GET['code']);  
  $\_SESSION[$tokenSessionKey] = $client->getAccessToken();  
  header('Location: ' . $redirect);  
}  
  
if (isset($\_SESSION[$tokenSessionKey])) {  
  $client->setAccessToken($\_SESSION[$tokenSessionKey]);  
}  
  
// Check to ensure that the access token was successfully acquired.  
if ($client->getAccessToken()) {  
  $htmlBody = '';  
  try{  
  
    // REPLACE this value with the video ID of the video being updated.  
    $videoId = "VIDEO\_ID";  
  
    // REPLACE this value with the path to the image file you are uploading.  
    $imagePath = "/path/to/file.png";  
  
    // Specify the size of each chunk of data, in bytes. Set a higher value for  
    // reliable connection as fewer chunks lead to faster uploads. Set a lower  
    // value for better recovery on less reliable connections.  
    $chunkSizeBytes = 1 \* 1024 \* 1024;  
  
    // Setting the defer flag to true tells the client to return a request which can be called  
    // with ->execute(); instead of making the API call immediately.  
    $client->setDefer(true);  
  
    // Create a request for the API's thumbnails.set method to upload the image and associate  
    // it with the appropriate video.  
    $setRequest = $youtube->thumbnails->set($videoId);  
  
    // Create a MediaFileUpload object for resumable uploads.  
    $media = new Google\_Http\_MediaFileUpload(  
        $client,  
        $setRequest,  
        'image/png',  
        null,  
        true,  
        $chunkSizeBytes  
    );  
    $media->setFileSize(filesize($imagePath));  
  
  
    // Read the media file and upload it chunk by chunk.  
    $status = false;  
    $handle = fopen($imagePath, "rb");  
    while (!$status && !feof($handle)) {  
      $chunk = fread($handle, $chunkSizeBytes);  
      $status = $media->nextChunk($chunk);  
    }  
  
    fclose($handle);  
  
    // If you want to make other calls after the file upload, set setDefer back to false  
    $client->setDefer(false);  
  
  
    $thumbnailUrl = $status['items'][0]['default']['url'];  
    $htmlBody .= "<h3>Thumbnail Uploaded</h3><ul>";  
    $htmlBody .= sprintf('<li>%s (%s)</li>',  
        $videoId,  
        $thumbnailUrl);  
    $htmlBody .= sprintf('<img src="%s">', $thumbnailUrl);  
    $htmlBody .= '</ul>';  
  
  
  } catch (Google\_Service\_Exception $e) {  
    $htmlBody .= sprintf('<p>A service error occurred: <code>%s</code></p>',  
        htmlspecialchars($e->getMessage()));  
  } catch (Google\_Exception $e) {  
    $htmlBody .= sprintf('<p>An client error occurred: <code>%s</code></p>',  
        htmlspecialchars($e->getMessage()));  
  }  
  
  $\_SESSION[$tokenSessionKey] = $client->getAccessToken();  
} elseif ($OAUTH2\_CLIENT\_ID == 'REPLACE\_ME') {  
  $htmlBody = <<<END  
  <h3>Client Credentials Required</h3>  
  <p>  
    You need to set <code>\$OAUTH2\_CLIENT\_ID</code> and  
    <code>\$OAUTH2\_CLIENT\_ID</code> before proceeding.  
  <p>  
END;  
} else {  
  // If the user hasn't authorized the app, initiate the OAuth flow  
  $state = mt\_rand();  
  $client->setState($state);  
  $\_SESSION['state'] = $state;  
  
  $authUrl = $client->createAuthUrl();  
  $htmlBody = <<<END  
<h3>Authorization Required</h3>  
<p>You need to <a href="$authUrl">authorize access</a> before proceeding.<p>  
END;  
}  
?>