This PROJECT HAS BEEN COMPLETED AND WORKING IN PHP. ☑️

-Eibil Yousibe…….  
  
  
  
Method: revisions.get

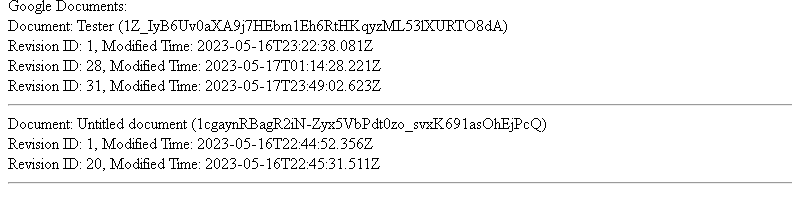
bookmark\_border

Gets a revision's metadata or content by ID.

HTTP request

GET https://www.googleapis.com/drive/v3/files/{fileId}/revisions/{revisionId}

The URL uses gRPC Transcoding syntax.



if successful, the response body contains an instance of [Revision](https://developers.google.com/drive/api/reference/rest/v3/revisions#Revision).

USING A DIFF(SEE DIFFERENCE between 2 Revisions(or DOWNLOAD 1 Version CONTROL TO SEE DIFFERENCE TO CURRENT VERSION[USING API]))

<https://stackoverflow.com/questions/36235099/compare-two-text-files-via-php-but-find-diffrence>

Or

<https://github.com/baraja-core/simple-php-diff> [

Downloading Revisions and doing a SIMPLE PHP DIFF

<https://developers.google.com/drive/api/guides/manage-revisions>

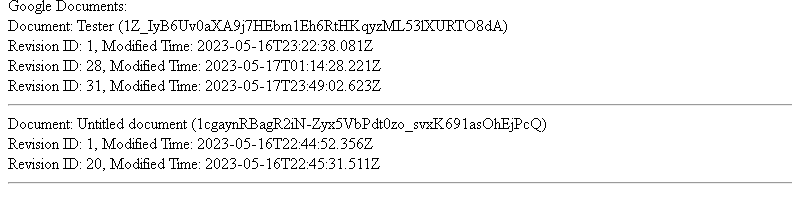
]

Using

<https://developers.google.com/drive/api/reference/rest/v3/revisions/get>

Google Meta Data and field of revision

<https://developers.google.com/drive/api/reference/rest/v3/revisions>

Download a revision You can use the value of the headRevisionId field from the Files resource to download the latest revision. This fetches the revision that corresponds to the metadata of the file you previously retrieved. To download the data for all previous revisions of the file that are still stored in cloud, you can call revisions.list, which gets all the revisionIds.

Revision ID:

| <?php require\_once 'vendor/autoload.php';  session\_start();  $client = new Google\_Client(); $client->setAuthConfig('client\_secrets.json'); // Path to your client\_secrets.json file $client->addScope(Google\_Service\_Drive::DRIVE\_METADATA\_READONLY); $client->addScope(Google\_Service\_Docs::DOCUMENTS\_READONLY); $client->setRedirectUri('http://' . $\_SERVER['HTTP\_HOST'] . '/oauth2callback.php'); // Path to your OAuth 2.0 callback file  if (isset($\_SESSION['access\_token']) && $\_SESSION['access\_token']) {  $client->setAccessToken($\_SESSION['access\_token']);  $driveService = new Google\_Service\_Drive($client);   // Query to get Google Documents owned by the user  $optParams = array(  'q' => "'me' in owners and mimeType='application/vnd.google-apps.document'",  'fields' => 'files(id, name)',  );   $results = $driveService->files->listFiles($optParams);   if (count($results->getFiles()) == 0) {  print "No Google Documents found.\n";  } else {  $userInfo = $driveService->about->get(['fields' => 'user']);  $googleUsername = $userInfo->getUser()->getDisplayName();   echo "Hello : " . $googleUsername . "<br><br>";   print "Google Documents:<br>";  foreach ($results->getFiles() as $file) {  printf("Document: %s (%s)<br>", $file->getName(), $file->getId());   $revisions = $driveService->revisions->listRevisions($file->getId());  foreach ($revisions->getRevisions() as $revision) {  $modTime = $revision->getModifiedTime();  echo "Revision ID: " . $revision->getId() . ", Modified Time: " . $modTime . "<br>";  }   echo "<hr>"; // Add a horizontal rule  }  } } else {  $redirect\_uri = 'http://' . $\_SERVER['HTTP\_HOST'] . '/oauth2callback.php'; // Path to your OAuth 2.0 callback file  header('Location: ' . filter\_var($redirect\_uri, FILTER\_SANITIZE\_URL)); } ?> |
| --- |

Explanation:

Google API Specifically GOOGLE DRIVE / GOOGLE DOCS

Allows for revision IDS to be collected. May not be incremented by 1.

This will allow to do a

| GET https://www.googleapis.com/drive/v3/files/{fileId}/revisions/{revisionId} |
| --- |

HTTPS Request to DOWNLOAD Revisions, and using a PHP Diff to see the difference in revisions.

NOTE - Already ALLOW API to see the current versions. Current Version Words can be seen by API but previous Version Histories to compare must be downloaded and a php diff initiated.

The Google Document Revision API allows you to download revisions of Google Docs files. Once you have downloaded the revisions, you can use a PHP diff tool to see the differences between them. A PHP diff tool is a program that can compare two text files and show the differences between them. There are many different PHP diff tools available, such as PHP\_Diff, Diff, and PHP\_Diff\_Ext.

Or Download highest revision ID to current(No need to download current version, already provided by API). Version History previous versions can only be downloaded.

ALL DOCUMENTATION USED

-MetaData / Fields

<https://developers.google.com/drive/api/reference/rest/v3/revisions#Revision>

* HTTPS REQ OR GET

<https://developers.google.com/drive/api/reference/rest/v3/revisions/get>

-DOWNLOAD REVISION

<https://developers.google.com/drive/api/guides/manage-revisions>

Example:NOT)WORKing

| <?php |
| --- |

| require\_once 'vendor/autoload.php';  session\_start();  $client = new Google\_Client(); $client->setAuthConfig('client\_secrets.json'); // Path to your client\_secrets.json file $client->addScope(Google\_Service\_Drive::DRIVE\_METADATA\_READONLY); $client->addScope(Google\_Service\_Docs::DOCUMENTS\_READONLY); $client->setRedirectUri('http://' . $\_SERVER['HTTP\_HOST'] . '/oauth2callback.php'); // Path to your OAuth 2.0 callback file  if (isset($\_SESSION['access\_token']) && $\_SESSION['access\_token']) {  $client->setAccessToken($\_SESSION['access\_token']);  $driveService = new Google\_Service\_Drive($client);   // Query to get Google Documents owned by the user  $optParams = array(  'q' => "'me' in owners and mimeType='application/vnd.google-apps.document'",  'fields' => 'files(id, name)',  );   $results = $driveService->files->listFiles($optParams);   if (count($results->getFiles()) == 0) {  print "No Google Documents found.\n";  } else {  $userInfo = $driveService->about->get(['fields' => 'user']);  $googleUsername = $userInfo->getUser()->getDisplayName();   echo "Hello : " . $googleUsername . "<br><br>";   print "Google Documents:<br>";  foreach ($results->getFiles() as $file) {  printf("Document: %s (%s)<br>", $file->getName(), $file->getId());   // Get the highest revision ID  $revisions = $driveService->revisions->listRevisions($file->getId());  $highestRevisionId = $revisions->getRevisions()[-1]->getId();   // Download the content of the revision  $revision = $driveService->revisions->get(fileId=file->getId(), revisionId=$highestRevisionId);  $content = $revision->getContent();   // Save the content of the revision to a file  $fileName = $file->getName() . "\_highest\_revision.txt";  with open($fileName, "wb") as f:  f.write($content);   echo "Downloaded highest revision of $fileName to $fileName.<br>";  }  } } else {  $redirect\_uri = 'http://' . $\_SERVER['HTTP\_HOST'] . '/oauth2callback.php'; // Path to your OAuth 2.0 callback file  header('Location: ' . filter\_var($redirect\_uri, FILTER\_SANITIZE\_URL)); } ?> |
| --- |

C#

| using System; using Google.Apis.Auth.OAuth2; using Google.Apis.Drive.v3; using Google.Apis.Drive.v3.Data; using Google.Apis.Services;  class Program {  static void Main(string[] args)  {  string[] scopes = { DriveService.Scope.DriveMetadataReadonly, DriveService.Scope.DocumentsReadonly };  string clientSecretPath = "client\_secrets.json"; // Path to your client\_secrets.json file  string redirectUri = "http://localhost:8080/oauth2callback"; // Path to your OAuth 2.0 callback file   GoogleClientSecrets secrets;  using (var stream = new System.IO.FileStream(clientSecretPath, System.IO.FileMode.Open, System.IO.FileAccess.Read))  {  secrets = GoogleClientSecrets.Load(stream);  }   var credential = GoogleWebAuthorizationBroker.AuthorizeAsync(  secrets.Secrets,  scopes,  "user",  System.Threading.CancellationToken.None,  new FileDataStore("Drive.Auth.Store")).Result;   var service = new DriveService(new BaseClientService.Initializer()  {  HttpClientInitializer = credential,  ApplicationName = "Your Application Name",  });   FilesResource.ListRequest listRequest = service.Files.List();  listRequest.Q = "'me' in owners and mimeType='application/vnd.google-apps.document'";  listRequest.Fields = "files(id, name)";  FileList files = listRequest.Execute();   if (files.Files.Count == 0)  {  Console.WriteLine("No Google Documents found.");  }  else  {  var userInfo = service.About.Get().Execute();  string googleUsername = userInfo.User.DisplayName;  Console.WriteLine("Hello: " + googleUsername + "\n");   Console.WriteLine("Google Documents:");  foreach (var file in files.Files)  {  Console.WriteLine("Document: {0} ({1})", file.Name, file.Id);   // Get the highest revision ID  RevisionsResource.ListRequest revisionsRequest = service.Revisions.List(file.Id);  RevisionList revisions = revisionsRequest.Execute();  string highestRevisionId = revisions.Revisions[revisions.Revisions.Count - 1].Id;   // Download the content of the revision  RevisionsResource.GetRequest revisionRequest = service.Revisions.Get(file.Id, highestRevisionId);  Revision revision = revisionRequest.Execute();  string content = revision.Content;   // Save the content of the revision to a file  string fileName = file.Name + "\_highest\_revision.txt";  System.IO.File.WriteAllText(fileName, content);   Console.WriteLine("Downloaded highest revision of {0} to {1}.", fileName, fileName);  }  }   Console.ReadKey();  } } |
| --- |

TO DO; UPDATED

1. Fetch the current version of the Google Document: You can fetch the content of the current version of the document using the Google Docs API. This can be achieved with the documents.get method. [✅](https://fsymbols.com/signs/tick/)
2. Read the content of your local DOCX file: Use phpoffice/phpword to read the content of the local DOCX file. This will allow you to extract the text from the file. ✅
3. Compare the contents: Now that you have the contents of both the current Google Doc version and the local DOCX file, you can compare them to find the differences. There are many ways to perform a text comparison; one common method is to use the Longest Common Subsequence (LCS) algorithm, which finds the longest subsequence common to all sequences in a set of sequences (often just two sequences). You might need to use or implement a PHP library that provides this functionality. ☑️

Now ALL that is needed to be done is CHECK CURRENT VERSION(ALREADY DONE) and check Untitled document.docx(DOWNLOADED FILE) 😀

\PhpOffice\PhpWord\IOFactory::load(

| <?php require\_once 'vendor/autoload.php';  $phpWord = \PhpOffice\PhpWord\IOFactory::load('Untitled document.docx');  $text = ''; foreach ($phpWord->getSections() as $section) {  $elements = $section->getElements();  foreach ($elements as $element) {  if (get\_class($element) === 'PhpOffice\PhpWord\Element\TextRun') {  foreach ($element->getElements() as $textElement) {  $text .= $textElement->getText();  }  }  } }  echo $text; // The content of the DOCX file ?> |
| --- |

Finally - Compare

Index.php- DOWNLOAD DocX

Index2.php - READ DOCX / PUT IMAGES IN FOLDER

Slideshow.php - Looks at spreadsheet and determines if A1-C10 has at least 5 words.

| $highestRevision = null;  foreach ($revisions->getRevisions() as $revision) {  if ($highestRevision === null || $revision->getId() > $highestRevision->getId()) {  $highestRevision = $revision;  }  }  YOU CAN MODIFY BUT WHAT I DID IS GET THE NEWEST REVISION TO THE CURRENT VERSION |
| --- |

You can fetch the revision history of the slide presentation using Google Drive API and then compare different revisions to see if there has been any changes in the 'pageElements' of the slides. If there has been a change, you can conclude that a transition or an animation might have been applied. But please note that 'pageElements' changes can also occur due to other factors like adding or deleting shapes, texts, etc. So, this isn't a reliable solution.

HISTORY REVISIONS DCS/ CHECK REVISIONS ON THE GOOGLE DOCUMENT ☑️

GOOGLE SLIDES / TRansition / Animation / ☑️

GOOGLE SHEETS / Chart USED IMG / A1-C10 (At Least 5 words in that range A1-c10{5-10) ☑️